

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2002-342165

(43)Date of publication of application : 29.11.2002

(51)Int.Cl.

G06F 12/14

G06F 15/16

(21)Application number : 2002-079358

(71)Applicant : SONY COMPUTER
ENTERTAINMENT INC

(22)Date of filing : 20.03.2002

(72)Inventor : SUZUOKI MASAKAZU
YAMAZAKI TAKESHI

(30)Priority

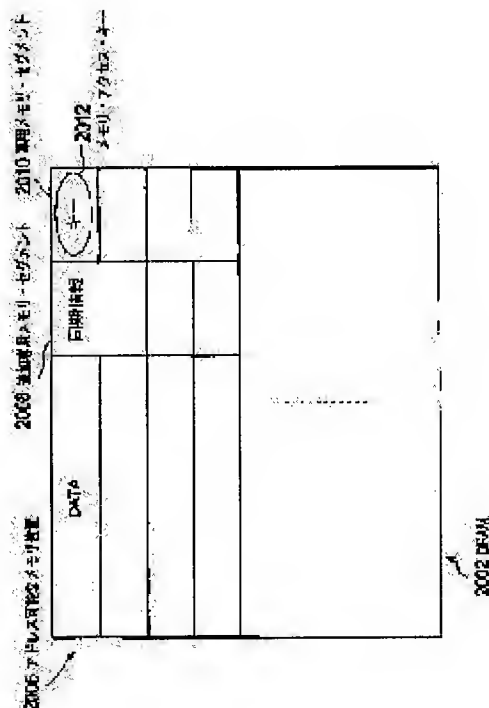
Priority number : 2001 816020 Priority date : 22.03.2001 Priority country : US

(54) COMPUTER PROCESSING SYSTEM AND PROCESSING METHOD PERFORMED IN
COMPUTER

(57)Abstract:

PROBLEM TO BE SOLVED: To provide a computer architecture and programming model for high speed processing over broadband networks.

SOLUTION: The architecture employs a consistent modular structure, a common computing module and uniform software cells. The common computing module includes a control processor, a plurality of processing units, a plurality of local memories from which the processing units process programs, a direct memory access, controller and a shared main memory. A synchronized system and method for the coordinated reading and writing of data to and from the shared main memory by the processing units also are provided.



(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開2002-342165

(P2002-342165A)

(43) 公開日 平成14年11月29日 (2002. 11. 29)

(51) Int. Cl. ⁷	識別記号	F I	特許庁(参考)
G 0 6 F 12/14	3 1 0	G 0 6 F 12/14	3 1 0 J 5 B 0 1 7
15/16	6 1 0	15/16	6 1 0 F 5 B 0 4 5
	6 2 0		6 2 0 B
			6 2 0 G

審査請求 有 請求項の数40 O L (全 40 頁)

(21) 出願番号 特願2002-79356(P2002-79356)

(22) 出願日 平成14年3月20日 (2002. 3. 20)

(31) 優先権主張番号 0 9 / 8 1 6 , 0 2 0

(32) 優先日 平成13年3月22日 (2001. 3. 22)

(33) 優先権主張国 米国 (US)

(71) 出願人 395015319

株式会社ソニー・コンピュータエンタテインメント

東京都港区赤坂7-1-1

(72) 発明者 鈴木 雅一

東京都港区赤坂7丁目1番1号 株式会社
ソニー・コンピュータエンタテインメント
内

(74) 代理人 100099324

弁理士 鈴木 正剛 (外2名)

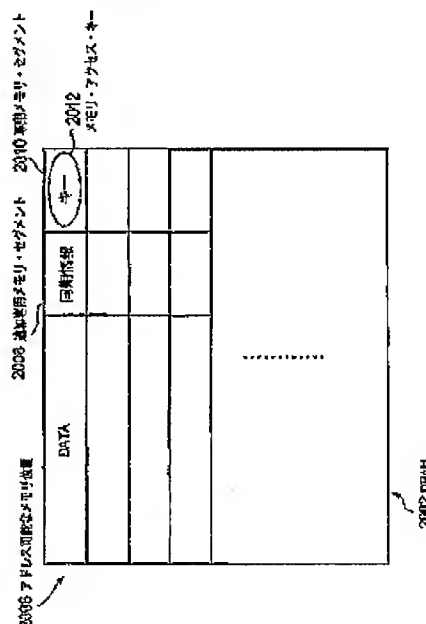
最終頁に続く

(54) 【発明の名称】 コンピュータ処理システム及びコンピュータで実行される処理方法

(57) 【要約】

【課題】 広帯域ネットワークを介する高速処理用コンピュータ・アーキテクチャとプログラミング・モデルが提供される。

【解決手段】 上記アーキテクチャは、均一なモジュール構造と、共通のコンピューティング・モジュールと、均一なソフトウェア・セルとを用いる。共通のコンピューティング・モジュールの中には、制御装置と、複数の処理ユニットと、処理ユニットがプログラムを処理する元となる複数のローカルメモリと、ダイレクト・メモリ・アクセスと、コントローラと、共用メイン・メモリとが含まれる。共用メイン・メモリからのデータの調整された読み出しと書き込みを処理ユニットによって行うための同期システムと方法とが提供される。



(2)

特開2002-342165

1

【特許請求の範囲】

【請求項1】 プログラムと前記プログラムと関連付けられたデータとを格納する第1メモリと、

前記プログラムとそれに関連付けられた前記データとを処理する複数の第1処理ユニットと、

前記第1処理ユニットによる前記第1メモリへのアクセスを制御するためのメモリ・コントローラと、

アクセス・テーブルとキー・テーブルを格納する第2メモリであって、前記アクセス・テーブルは、複数のアクセス・エントリを有し、各アクセス・エントリは、アクセス・キーと、前記アクセス・キーと関連付けられた前記第1メモリ内の1つのメモリ・スペースの識別子を含み、前記キー・テーブルは、複数のキー・エントリを有し、各キー・エントリは、前記第1処理ユニットの識別子と前記第1処理ユニットと関連するリクエスト・キーを含んで成る第2メモリと、

前記第1処理ユニットによる前記プログラムとそれに関連付けられた前記データの処理を制御するための第2処理ユニットであって、前記アクセス・テーブルと前記キー・テーブルの作成と更新の操作が可能で、更に、前記第1処理ユニットによる前記プログラムの1つの処理を行う指示を出すことが可能となる前記第2処理ユニットとを有し、

前記第1処理ユニットの1つは、前記1つのプログラムの処理において前記メモリ・コントローラにリクエストを出して、前記第1メモリ内の格納位置にアクセスすることが可能であり、

前記メモリ・コントローラは、前記リクエストに応答して、前記キー・テーブル内の前記第1処理ユニットの1つと関連付けられたリクエスト・キーと、前記アクセス・テーブル内のアクセス・キーとを比較して、前記アクセス・キーの1つが、前記キー・テーブル内の前記第1処理ユニットの1つと関連付けられたリクエスト・キーと一致して、かつ、前記格納位置が、前記アクセス・テーブル内の前記1つのアクセス・キーと関連付けられたメモリ・スペースと一致する場合は、前記リクエストの実行が可能であることを特徴とするコンピュータ処理システム。

【請求項2】 請求項1に記載のコンピュータ処理システムにおいて、複数のローカル・メモリをさらに有し、各々の前記ローカル・メモリは、前記第1処理ユニットの1つと関連付けられていて、前記第2処理ユニットが、前記メモリ・コントローラに対して、前記第1メモリから、前記1つの第1処理ユニットに関連付けられたローカル・メモリへ、前記1つのプログラムの転送指示を出すことによって、前記1つの第1処理ユニットに前記1つのプログラムの処理を指示することが可能であり、その後、前記1つの第1処理ユニットが、前記ローカル・メモリの前記1つのプログラムを処理することを特徴とするコンピュータ処理システム。

2

【請求項3】 請求項1に記載のコンピュータ処理システムにおいて、前記1つのアクセス・キーが、第1の複数ビットを有し、前記1つの第1処理ユニットに関連付けられた前記リクエスト・キーが、第2の複数ビットを有し、前記第1の複数ビットのすべてと、前記第2の複数ビットのすべてとが一致する場合にのみ、前記メモリ・コントローラが、前記リクエストを実行可能であることを特徴とするコンピュータ処理システム。

【請求項4】 請求項1に記載のコンピュータ処理システムにおいて、前記1つのアクセス・キーが、第1の複数ビットとキー・マスクとを有し、前記第1処理ユニットに関連付けられた前記リクエスト・キーが、第2の複数ビットを有し、前記第1の複数ビットのすべてが、前記第2の複数ビットのすべてと一致する場合、または前記第2の複数ビットに一致しない前記第1の複数ビットのすべてが前記マスクによってマスクされている場合のいずれかにおいてのみ、前記メモリ・コントローラが、前記リクエストを実行可能であることを特徴とするコンピュータ処理システム。

【請求項5】 請求項1に記載のコンピュータ処理システムにおいて、前記1つのアクセス・キーが、第1の複数ビットを有し、前記第1処理ユニットに関連付けられた前記リクエスト・キーが、第2の複数ビットとキー・マスクとを有し、前記第1の複数ビットのすべてが、前記第2の複数ビットのすべてと一致する場合、または前記第1の複数ビットに一致しない前記第2の複数ビットのすべてが前記マスクによってマスクされている場合のいずれかにおいてのみ、前記メモリ・コントローラが、前記リクエストを実行可能であることを特徴とするコンピュータ処理システム。

【請求項6】 請求項1に記載のコンピュータ処理システムにおいて、各々の前記キー・エントリが、キー・マスクをさらに含むことを特徴とするコンピュータ処理システム。

【請求項7】 請求項1に記載のコンピュータ処理システムにおいて、各々の前記アクセス・エントリが、キー・マスクをさらに含むことを特徴とするコンピュータ処理システム。

【請求項8】 請求項1に記載のコンピュータ処理システムにおいて、各々の前記キー・エントリが、キー・マスクをさらに含み、各々の前記アクセス・エントリが、キー・マスクをさらに含むことを特徴とするコンピュータ処理システム。

【請求項9】 請求項1に記載のコンピュータ処理システムにおいて、各々の前記アクセス・エントリが、ベース位置エントリとサイズ・エントリをさらに含み、前記ベース位置エントリが、前記アクセス・キーに関連付けられた前記メモリ・スペース内の前記第1メモリ内の開始アドレスを指定し、前記サイズ・エントリが、前記アクセス・エントリに関連付けられた前記メモリ・スパー

(3)

特開2002-342165

3

スの前記第1メモリ内のサイズを指定することを特徴とするコンピュータ処理システム。

【請求項10】 請求項1に記載のコンピュータ処理システムにおいて、前記第1メモリが、ダイナミック・ランダム・アクセス・メモリであって、かつ、前記コンピュータ処理システムのメイン・メモリであることを特徴とするコンピュータ処理システム。

【請求項11】 コンピュータで実行される処理方法において、

第1メモリに、プログラムと、前記プログラムと関連付けられたデータと、を格納するステップと、

複数の第1処理ユニットを用いて、前記プログラムとそれに関連付けられた前記データとを処理するステップと、

メモリ・コントローラを用いて、前記第1処理ユニットによる前記第1メモリへのアクセスを制御するステップと、

第2処理ユニットを用いて、第2メモリに、アクセス・テーブルとキー・テーブルを作成するステップであって、前記アクセス・テーブルは、複数のアクセス・エントリを有し、各アクセス・エントリが、前記アクセス・キーと、前記アクセス・キーと関連付けられた前記第1メモリのメモリ・スペースの識別子を含み、前記キー・テーブルが、複数のキー・エントリを有し、各キー・エントリが、1つの前記第1処理ユニットの識別子と、前記第1処理ユニットと関連付けられたリクエスト・キーを含むように構成するステップと、

前記第2処理ユニットを用いて、前記第1処理ユニットによる前記プログラムとそれに関連付けられた前記データの処理を制御するステップと、

前記第2処理ユニットを用いて、前記第1処理ユニットのうちの1つに、前記プログラムのうちの1つの処理を命令するステップと、前記1つの第1処理ユニットから、前記1つのプログラムの処理において、前記メモリ・コントローラにリクエストを出して、前記第1メモリ内の格納位置へアクセスするステップと、

前記リクエストに回答して、前記1つの第1処理ユニットと関連付けられたリクエスト・キーと、前記アクセス・テーブル内の前記アクセス・キーとを比較するステップと、

1つの前記アクセス・キーが、前記キー・テーブル内の前記1つの処理ユニットと関連付けられたリクエスト・キーと一致し、前記格納位置が、前記アクセス・テーブル内で指定された前記1つのアクセス・キーと関連付けられたメモリ・スペースと一致する場合は、前記リクエストを実行するステップと、を有することを特徴とする方法。

【請求項12】 請求項11に記載の方法において、前記第1メモリから前記第1処理ユニットに関連付けられたローカル・メモリへ前記1つのプログラムを転送する

4

ように前記第2処理ユニットから、前記メモリ・コントローラにコマンドを出すことによって、前記第2処理ユニットを用いて、前記1つの第1処理ユニットに対して前記1つのプログラムの処理を命令し、その後、前記第1処理ユニットを用いて、前記ローカル・メモリから前記1つのプログラムを処理するステップをさらに有することを特徴とする方法。

【請求項13】 請求項11に記載の方法において、前記1つのアクセス・キーが、第1の複数のビットを有し、前記第1処理ユニットに関連付けられた前記リクエスト・キーが、第2の複数のビットを有し、

前記第1の複数のビットのすべてが、前記第2の複数のビットのすべてと一致する場合にのみ、前記メモリ・コントローラが、前記リクエストを実行するステップを、さらに有することを特徴とする方法。

【請求項14】 請求項11に記載の方法において、前記1つのアクセス・キーが、第1の複数のビットとキー・マスクとを有し、前記第1処理ユニットに関連付けられた前記リクエスト・キーが、第2の複数のビットを有し、前記第2の複数のビットのすべてが、前記第1の複数のビットのすべてと一致する場合、または前記第2の複数のビットに一致しない前記第1の複数のビットのすべてが前記マスクによってマスクされている場合のいずれかにおいてのみ、前記メモリ・コントローラが、前記リクエストを実行するステップをさらに有することを特徴とする方法。

【請求項15】 請求項11に記載の方法において、前記1つのアクセス・キーが、第1の複数のビットを有し、前記第1処理ユニットに関連付けられた前記リクエスト・キーが、第2の複数のビットとキー・マスクとを有し、前記第1の複数のビットのすべてが、前記第2の複数のビットのすべてと一致する場合、または前記第1の複数のビットに一致しない前記第2の複数のビットのすべてが前記マスクによってマスクされている場合のいずれかにおいてのみ、前記メモリ・コントローラが、前記リクエストを実行するステップをさらに有することを特徴とする方法。

【請求項16】 請求項11に記載の方法において、各々の前記キー・エントリが、キー・マスクをさらに含むことを特徴とする方法。

【請求項17】 請求項11に記載の方法において、各々の前記アクセス・エントリが、キー・マスクをさらに含むことを特徴とする方法。

【請求項18】 請求項11に記載の方法において、各々の前記キー・エントリが、キー・マスクをさらに含み、各々の前記アクセス・エントリが、キー・マスクをさらに含むことを特徴とする方法。

【請求項19】 請求項11に記載の方法において、各々の前記アクセス・エントリが、ベース位置エントリとサイズ・エントリをさらに含み、前記ベース位置エントリ

(4)

特開2002-342165

5

りが、前記アクセス・キーに関連付けられた前記メモリ・スペースの前記第1メモリ内の前記開始アドレスを指定し、前記サイズ・エントリが、前記アクセス・エントリに関連付けられた前記メモリ・スペースの前記第1メモリ内のサイズを指定することを特徴とする方法。

【請求項20】 請求項11に記載の方法において、前記第1メモリが、ダイナミック・ランダム・アクセス・メモリであって、かつ、前記コンピュータ処理システムのメイン・メモリであることを特徴とする方法。

【請求項21】 プログラムと前記プログラムと関連付けられたデータを格納する第1メモリであって、複数のアドレス可能な格納位置を有し、アドレス可能な各格納位置は、前記アドレス可能な格納位置に関連付けられるとともに前記アドレス可能な格納位置のアクセス・キーを含む追加メモリ・セグメント有して成る第1メモリと、

前記プログラムとそれに関連付けられた前記データとを処理するための複数の第1処理ユニットと、前記第1メモリへの前記第1処理ユニットによるアクセス制御のためのメモリ・コントローラと、複数のキー・エントリから成るキー・テーブルを格納するための第2メモリであって、各キー・エントリが、前記第1処理ユニットの1つの識別子と、前記第1処理ユニットと関連付けられたリクエスト・キーとを含んでなる第2メモリと、

前記第1処理ユニットによる前記プログラムとそれに関連付けられた前記データの処理を制御するための第2処理ユニットであって、前記アクセス・キーの割り当てと更新および前記アクセス・キー・テーブルの作成と更新が可能で、更に、前記第1処理ユニットのうちの1つによる前記プログラムの1つの処理を指示することが可能となる第2処理ユニットとを有し、

前記第1処理ユニットの1つは、前記プログラムの1つの処理において前記メモリ・コントローラにリクエストを出して、前記第1メモリ内の格納位置をアクセスすることが可能であり、

前記メモリ・コントローラは、前記リクエストに応答して、前記キー・テーブル内の前記第1処理ユニットの1つと関連付けられたリクエスト・キーと、前記1つのアドレス可能な格納位置と関連付けられている追加メモリ・セグメントに含まれているアクセス・キーとを比較して、前記キー・テーブル内の前記第1処理ユニットの1つと関連付けられたリクエスト・キーと、前記1つのアドレス可能な格納位置と関連付けられている追加メモリ・セグメントに含まれている前記アクセス・キーが一致する場合は、前記リクエストを実行することが可能であることを特徴とするコンピュータ処理システム。

【請求項22】 請求項21に記載のコンピュータ処理システムにおいて、複数のローカル・メモリをさらに有し、前記ローカル・メモリの各々は、前記第1処理ユニ

6

ットの1つと関連付けられていて、前記第2処理ユニットが、前記メモリ・コントローラに対して、前記第1メモリから前記1つのプログラムを前記第1処理ユニットに関連付けられたローカル・メモリへ転送する指示を出すことによって、前記1つの第1処理ユニットに前記プログラムの処理を指示することが可能であり、その後、前記1つの第1処理ユニットが、前記ローカル・メモリから前記1つのプログラムを処理可能であることを特徴とするコンピュータ処理システム。

【請求項23】 請求項21に記載のコンピュータ処理システムにおいて、前記1つのアドレス可能なメモリ位置に関連付けられた前記メモリ・セグメントに含まれている前記アクセス・キーが、第1の複数のビットを有し、前記第1処理ユニットに関連付けられた前記リクエスト・キーが、第2の複数のビットを有し、前記第1の複数のビットのすべてが、前記第2の複数のビットのすべてと一致する場合にのみ、前記メモリ・コントローラが、前記リクエストを実行可能であることを特徴とするコンピュータ処理システム。

【請求項24】 請求項21に記載のコンピュータ処理システムにおいて、前記1つのアドレス可能なメモリ位置に関連付けられた前記メモリ・セグメントに含まれている前記アクセス・キーが、第1の複数のビットとキー・マスクを有し、前記第1処理ユニットに関連付けられた前記リクエスト・キーが、第2の複数のビットを有し、前記第1の複数のビットのすべてが、前記第2の複数のビットのすべてと一致する場合、または前記第2の複数のビットに一致しない前記第1の複数のビットのすべてが前記マスクによってマスクされている場合のいずれかにおいてのみ、前記メモリ・コントローラが、前記リクエストを実行可能であることを特徴とするコンピュータ処理システム。

【請求項25】 請求項21に記載のコンピュータ処理システムにおいて、前記1つのアドレス可能な格納位置に関連付けられた前記メモリ・セグメントに含まれている前記アクセス・キーが、第1の複数のビットを有し、前記第1処理ユニットに関連付けられた前記リクエスト・キーが、第2の複数のビットとキー・マスクを有し、前記第1の複数のビットのすべてが、前記第2の複数のビットのすべてと一致する場合、または前記第1の複数のビットに一致しない前記第2の複数のビットのすべてが前記マスクによってマスクされている場合のいずれかにおいてのみ、前記メモリ・コントローラが、前記リクエストを実行可能であることを特徴とするコンピュータ処理システム。

【請求項26】 請求項21に記載のコンピュータ処理システムにおいて、各々の前記キー・エントリが、キー・マスクをさらに含むことを特徴とするコンピュータ処理システム。

【請求項27】 請求項21に記載のコンピュータ処理

(5)

特開2002-342165

7

システムにおいて、各々の前記追加メモリ・セグメントが、キー・マスクをさらに含むことを特徴とするコンピュータ処理システム。

【請求項28】 請求項21に記載のコンピュータ処理システムにおいて、各々の前記キー・エントリが、キー・マスクをさらに含み、各々の前記追加メモリ・セグメントが、キー・マスクをさらに含むことを特徴とするコンピュータ処理システム。

【請求項29】 請求項21に記載のコンピュータ処理システムにおいて、各々の前記追加メモリ・セグメントが、前記追加メモリ・セグメントに関連付けられたアドレス可能な格納位置に格納されているデータのステータスに関するステータス情報をさらに含むことを特徴とするコンピュータ処理システム。

【請求項30】 請求項21に記載のコンピュータ処理システムにおいて、前記第1メモリが、ダイナミック・ランダム・アクセス・メモリであって、かつ、前記コンピュータ処理システムのメイン・メモリであることを特徴とするコンピュータ処理システム。

【請求項31】 コンピュータ処理システムにてなされる処理方法において、

第1メモリに、プログラムと前記プログラムの関連データを格納するステップであって、複数のアドレス可能な格納位置を有し、前記アドレス可能な各格納位置には、前記アドレス可能な格納位置に関連付けられた追加メモリ・セグメントを含むステップと、

各々の前記アドレス可能な格納位置用の各々の前記追加メモリ・セグメントに、前記アドレス可能な格納位置のためのアクセス・キーを格納するステップと、

複数の第1処理ユニットを用いて、前記プログラムとそれに関連付けられた前記データを処理するステップと、メモリ・コントローラを用いて、前記第1処理ユニットによる前記第1メモリへのアクセスを制御するステップと、

第2メモリに、複数のキー・エントリから成るキー・テーブルを格納するステップであって、各キー・エントリが、前記第1処理ユニットと関連付けられた識別子と、前記第1処理ユニットと関連付けられたリクエスト・キーを含むステップと、

前記第2処理ユニットを用いて、前記第1処理ユニットによる前記プログラムとそれに関連付けられた前記データの処理を制御するステップと、

前記第2処理ユニットを用いて、各々の前記アクセス・キーを割り当てるステップと、

前記第2処理ユニットを用いて、前記キー・テーブルを作成するステップと、

前記第2処理ユニットを用いて、前記第1処理ユニットのいずれか1つに、前記プログラムの1つを実行する命令を出すステップと、

前記第1処理ユニットの1つから、前記1つのプログラ

8

ムの処理において、1つの前記アドレス可能な格納位置へアクセスするように、前記メモリ・コントローラへリクエストを出すステップと、

前記リクエストに回答して、前記キー・テーブル内で前記1つの第1処理ユニットと関連付けられたリクエスト・キーと、前記1つのアドレス可能な格納位置に格納されているアクセス・キーとを比較するステップと、

前記キー・テーブル内の前記1つの処理ユニットと関連付けられたリクエスト・キーと前記アドレス可能な格納位置と関連付けられた前記追加メモリ・セグメント内に格納されている前記アクセス・キーが一致する場合は、前記リクエストを実行するステップと、を有することを特徴とする方法。

【請求項32】 請求項31に記載の方法において、前記第2処理ユニットを用いて、前記第1メモリから前記1つの第1処理ユニットに関連付けられたローカル・メモリへ前記1つのプログラムを転送するように前記メモリ・コントローラに命令することによって、前記1つの第1処理ユニットに対して前記1つのプログラムの処理を命令するステップと、その後、前記第1処理ユニットによって、前記ローカル・メモリから前記1つのプログラムを実行するステップと、をさらに有することを特徴とする方法。

【請求項33】 請求項31に記載の方法において、前記1つのアドレス可能なメモリ位置に関連付けられた前記追加メモリ・セグメントに含まれている前記アクセス・キーが、第1の複数ビットを有し、前記第1処理ユニットに関連付けられた前記リクエスト・キーが、第2の複数ビットを有し、前記第1の複数ビットのすべてが、前記第2の複数ビットのすべてと一致する場合にのみ、前記メモリ・コントローラが、前記リクエストを実行するステップをさらに有することを特徴とするコンピュータ処理方法。

【請求項34】 請求項31に記載の方法において、前記1つのアドレス可能なメモリ位置に関連付けられた前記追加メモリ・セグメントに含まれている前記アクセス・キーが、第1の複数ビットとキー・マスクを有し、前記第1処理ユニットに関連付けられた前記リクエスト・キーが、第2の複数ビットを有し、前記第1の複数ビットのすべてが、前記第2の複数ビットのすべてと一致する場合、または前記第2の複数ビットに一致しない前記第1の複数ビットのすべてが前記マスクによってマスクされている場合のいずれかにおいてのみ、前記メモリ・コントローラが、前記リクエストを実行するステップをさらに有することを特徴とするコンピュータ処理方法。

【請求項35】 請求項31に記載の方法において、前記1つのアドレス可能なメモリ位置に関連付けられた前記追加メモリ・セグメントに含まれている前記アクセス・キーが、第1の複数ビットを有し、前記1つの第1処理ユニットに関連付けられた前記リクエスト・キーが、

(6)

特開2002-342165

9

10

第2の複数ビットとキー・マスクを有し、前記第2の複数ビットのすべてが、前記第1の複数ビットのすべてと一致する場合、または前記第1の複数ビットに一致しない前記第2の複数ビットのすべてが前記マスクによってマスクされている場合のいずれかにおいてのみ、前記メモリ・コントローラが、前記リクエストを実行するステップをさらに有することを特徴とするコンピュータ処理方法。

【請求項36】 請求項31に記載の方法において、各々の前記キー・エントリが、キー・マスクをさらに含むことを特徴とするコンピュータ処理方法。

【請求項37】 請求項31に記載の方法において、各々の前記追加メモリ・セグメントが、キー・マスクをさらに含むことを特徴とするコンピュータ処理方法。

【請求項38】 請求項31に記載の方法において、各々の前記キー・エントリが、キー・マスクをさらに含む、各々の前記追加メモリ・セグメントが、キー・マスクをさらに含むことを特徴とするコンピュータ処理方法。

【請求項39】 請求項31に記載の方法において、各々の前記追加メモリ・セグメントが、前記追加メモリ・セグメントに関連付けられたアドレス可能格納位置に格納されているデータの前記ステータスに関するステータス情報をさらに含むことを特徴とするコンピュータ処理方法。

【請求項40】 請求項31に記載の方法において、前記第1メモリが、ダイナミック・ランダム・アクセス・メモリであって、かつ前記コンピュータ処理システムのメイン・メモリであることを特徴とするコンピュータ処理方法。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】 本発明はコンピュータ・プロセッサ用アーキテクチャとコンピュータ・ネットワークとに関し、広帯域環境におけるコンピュータ・プロセッサおよびコンピュータ・ネットワーク用アーキテクチャに関する。さらに、本発明は、このようなアーキテクチャのためのプログラミング・モデルに関する。

【0002】

【従来の技術】 コンピュータおよび現今のコンピュータ・ネットワーク(オフィスのネットワークで使用するローカル・エリア・ネットワーク(LAN)やインターネットのようなグローバルネットワークなど)の計算用装置は、スタンド・アローン型の計算用として主として設計されてきた。コンピュータ・ネットワークを介するデータとアプリケーション・プログラム(“アプリケーション”)の共用は、これらのコンピュータおよびコンピュータ・デバイスの主要な設計目標ではなかった。これらのコンピュータとコンピュータ・デバイスは、また、様々な異なるメーカー(モトローラ、イ

ンテル、テキサス・インスツルメント、ソニーなど)によって製造された広範囲の異なるタイプのプロセッサを用いて一般に設計されたものである。これらのプロセッサの各々は、それ自身の特定の命令セットと命令セット・アーキテクチャ(ISA)とを持っている。すなわち、それ自身の特定のセットのアセンブリ言語命令と、これらの命令を実行する主演算装置と記憶装置のための構造とを有する。プログラマは、各プロセッサの命令セットとISAとを理解してこれらのプロセッサ用のアプリケーションを書くことを要求される。今日のコンピュータ・ネットワーク上でのコンピュータとコンピュータ・デバイスには異なった種類が混在していることから、データとアプリケーションの共用及びその処理は複雑になっている。さらに、この複数種が混在する環境に対する調整を行うために、多くの場合、同じアプリケーションであっても複数のバージョンを用意することが必要となっている。

【0003】 グローバルネットワーク、特にインターネットと接続されたタイプのコンピュータやコンピュータ・デバイスには広範囲に及ぶ。パーソナル・コンピュータ(PC)とサーバーに加えて、これらのコンピュータ・デバイスの中にはセルラー電話、移動用コンピュータ、個人用情報機器(PDA)、セット・トップ・ボックス、デジタルテレビ並びにその他の装置が含まれる。コンピュータやコンピュータ・デバイスにおいて異種製品が混在する中でデータやアプリケーションを共用することに起因して、大きな問題が生じている。

【0004】 これらの問題を解決するためのいくつかの手法が試みられてきた。これらの手法の中には、特に、優れたインターフェースと複雑なプログラミング手法が含まれる。これらの解決方法では、処理パワーの実質的増加の実現がしばしば要求される。また、これらの解決方法では、アプリケーションの処理に必要な時間と、ネットワークを介するデータ伝送に必要な時間とが実質的に増加してしまうという結果がしばしば生じる。

【0005】 一般に、データは、対応するアプリケーションとは別個に、インターネットを介して伝送される。このアプローチでは、アプリケーションに対応した各セットの伝送データにアプリケーション自体をも送る必要はなくなっている。従って、このアプローチによって、必要とされる帯域幅の量は最少化されるものの、ユーザーには不満の原因となることも多々ある。つまり、クライアント側のコンピュータでは、この伝送データを利用するための適正なアプリケーション、あるいは最新のアプリケーションを入手できない事態も生じ得る。また、このアプローチでは、ネットワーク上のプロセッサによって用いられている複数の異種ISAと命令セットに対応して、各アプリケーション毎にバージョンの異なる複数のアプリケーションを用意することが要求される。

(7)

特開2002-342165

11

【0006】Java（登録商標）モデルでは、この問題の解決が試みられている。このモデルでは、厳しいセキュリティ・プロトコルに準拠する小さなアプリケーション（“アプレット（applet）”）が用いられている。アプレットは、ネットワークを介してサーバー側コンピュータから送信されてクライアント側コンピュータ（“クライアント”）によって実行される。異なるISAを使用しているクライアント毎に、同じアプレットであっても異なるバージョンを送信するという事態を避ける必要があるため、すべてのJavaアプレットは、クライアント側のJava仮想マシン上で実行される。Java仮想マシンとは、JavaISAと命令セットとを備えたコンピュータをエミュレートするソフトウェアである。しかし、このソフトウェアは、クライアント側のISAとクライアント側の命令セットにより実行される。クライアント側ではISAと命令セットが各々異なるが、与えられるJavaの仮想マシンのバージョンは一つである。したがって、複数の各アプレットごとに異なるバージョンを用意する必要はない。各クライアントでは、当該クライアントにおけるISAと命令セットに
20 対応した適正なJava仮想マシンだけをダウンロードすれば、Javaアプレットを実行できる。

【0007】

【発明が解決しようとする課題】各々の異なるISAと命令セットに対して異なるバージョンのアプリケーションを書かなければならないという課題は解決されているものの、Javaの処理モデルでは、クライアント側のコンピュータに対してソフトウェアの追加層が要求される。ソフトウェアのこの追加層のためにプロセッサの処理速度は著しく低下する。この速度の低下は、リアルタイムのマルチメディア・アプリケーションについて特に著しい。また、ダウンロードされたJavaアプレットの中にはウィルス、処理上の誤動作などが含まれている可能性がある。これらのウィルスと誤動作はクライアントのデータベースの破損やその他の損害の原因となる可能性がある。Javaモデルで用いられているセキュリティ用プロトコルでは、“サンドボックス（sandbox）”（Javaアプレットがそれ以上はデータを書き込むことができない、クライアント側のメモリ内のスペース）というソフトウェアを設けることにより、この問題の解決が試みられているとはいえ、このソフトウェア駆動型セキュリティ・モデルはその実行時に頻繁に不安定になり、より多くの処理が必要となる。

【0008】リアルタイムの、マルチメディア・ネットワーク用アプリケーションがますます重要なものになりつつある。これらのネットワーク用アプリケーションでは非常に高速な処理が要求される。将来、そのようなアプリケーション用として毎秒何千メガビットものデータが必要となるかもしれない。ネットワークの現今のアーキテクチャ、および、特にインターネットのアーキテク

12

チャ、並びに、Javaモデルなどで現在実施されているプログラミング・モデルでこのような処理速度に到達することは非常に難しい。

【0009】したがって、新しいコンピュータ・アーキテクチャと、コンピュータ・ネットワーク用の新しいアーキテクチャと、新しいプログラミング・モデルとが求められている。この新しいアーキテクチャとプログラミング・モデルとによって、計算上の負担が付加されることがなく、ネットワークの様々なメンバー間でのデータとアプリケーションの共用という問題が解決されることが望ましい。また、この新しいコンピュータ・アーキテクチャと、プログラミング・モデルとによって、ネットワークのメンバー間でのアプリケーションとデータの共用時に生じる、固有のセキュリティ上の問題も解決されることが望ましい。

【0010】

【課題を解決するための手段】本発明の一実施形態においては、コンピュータと、コンピューティング・デバイスと、コンピュータ・ネットワーク（あるいはコンピュータ・ネットワークに代えて、コンピュータ・ネットワーク・システムや複数のコンピュータを備えたコンピュータ・システムというカテゴリーや形態とすることもできる）とのための新しいアーキテクチャが提供される。他の実施形態では、本発明は、これらのコンピュータ、コンピューティング・デバイスおよびコンピュータ・ネットワークのための新しいプログラミング・モデルを提供するものである。

【0011】本発明によれば、コンピュータ・ネットワークのすべてのメンバー（ネットワーク上のすべてのコンピュータとコンピューティング・デバイス）は共通のコンピューティング・モジュールから構成される。この共通のコンピューティング・モジュールは均一な構造を有し、好適には同じISAが使用される。ネットワークのメンバーとして、クライアント、サーバー、PC、移動用コンピュータ、ゲーム用マシン、PDA、セット・トップ・ボックス、電気機器、デジタルテレビ、および、コンピュータ・プロセッサを用いるその他の装置が挙げられる。均一なモジュラー構造によって、ネットワークのメンバーによるアプリケーションとデータの効率的な高速処理と、ネットワークを介するアプリケーションとデータの高遠伝送とが可能となる。またこの構造によって、様々なサイズと処理パワーを持つネットワークのメンバーの構成が単純化され、これらのメンバーによる処理用アプリケーションの作成が単純化される。

【0012】また、本発明によれば、コンピュータ・ネットワークにおいて、前記ネットワークと接続された複数のプロセッサを有し、前記プロセッサの各々が、同じ命令セット・アーキテクチャを有する複数の第1の処理ユニットと、前記第1の処理ユニットを制御するための第2の処理ユニットとを有し、前記第1の処理ユニット

(8)

特開2002-342165

13

が、前記ネットワークを介して伝送されるソフトウェア・セルを処理するために作動可能であり、前記ソフトウェア・セルの各々が、前記命令セット・アーキテクチャと互換性のあるプログラムと、前記プログラムと関連付けられたデータと、前記ネットワークを介して伝送される前記ソフトウェア・セルのすべての間で前記ソフトウェア・セルを一意的に識別するための識別子（例えばセルの識別番号）と、を有することを特徴とするコンピュータ・ネットワークも提供される。

【0013】なお、本発明によれば、コンピュータ・ネットワークと接続される複数のプロセッサを有するコンピュータ・システムであって、前記プロセッサの各々が、同じ命令セット・アーキテクチャを有する複数の第1の処理ユニットと、前記第1の処理ユニットを制御するための第2の処理ユニットとを有し、前記第1の処理ユニットが、前記ネットワークを介して伝送されるソフトウェア・セルを処理するために作動可能であり、前記ソフトウェア・セルの各々が、前記命令セット・アーキテクチャと互換性のあるプログラムと、前記プログラムと関連付けられたデータと、前記ネットワークを介して伝送される前記ソフトウェア・セルのすべての間で前記ソフトウェア・セルを一意的に識別するための識別子（例えばセルの識別番号）と、を有することを特徴とするコンピュータ・システムも提供される。

【0014】加えて、本発明によれば、コンピュータ・ネットワークを介する伝送用ソフトウェア・セルのデータ・ストリームにおいて、前記コンピュータ・ネットワークが、複数のプロセッサを有し、前記ソフトウェア・セルの各々が、前記プロセッサの中の1以上によって処理するためのプログラムと、前記プログラムと関連付けられたデータと、前記ネットワークを介して伝送されるすべてのソフトウェア・セルの中で前記ソフトウェア・セルを一意的に識別するグローバルな識別子と、を有することを特徴とするデータ・ストリームも提供される。なお、上記構成において、「データ・ストリーム」という形態に代えて、「データ構造」という形態、あるいは「上述のような構造を有するデータ」という形態で本発明を提供することも可能である。

【0015】他の実施形態では、本発明は、ネットワークを介してデータとアプリケーションを伝送するため、また、ネットワークのメンバー間でデータとアプリケーションを処理するための新しいプログラミング・モデルを提供する。このプログラミング・モデルでは、ネットワークのいずれのメンバーでも処理できる、ネットワークを介して伝送されるソフトウェア・セルが使用される。各ソフトウェア・セルは同じ構造を有し、アプリケーションとデータの双方を含むことが可能である。モジュラー型コンピュータ・アーキテクチャによって提供される高速処理と伝送速度の結果、これらのセルの高速処理が可能となる。アプリケーション用コードは同じ共

14

通の命令セットとISAに好適に基づくものである。各ソフトウェア・セルは、グローバルな識別子（グローバルID）と、セルの処理に必要な計算用リソースの量について説明する情報とを好適に含むことが望ましい。すべての計算用リソースは同じ基本構造を有し、同じISAが用いられているので、このセルの処理を実行する特定のリソースは、ネットワーク上のどこにでも配置が可能となり、動的に割り当てることができる。

【0016】基本となる処理用モジュールはプロセッサ・エレメント（PE）である。PEは、好適には、処理ユニット（PU）、ダイレクト・メモリ・アクセス・コントローラ（DMAC）および複数の付加処理ユニット（APU）を具備することが望ましい。好ましい実施形態では、1つのPEは8つのAPUを具備する。PUとAPUとは、クロスバ・アーキテクチャを好適に備えている共用ダイナミック・ランダム・アクセス・メモリ（DRAM）を用いてリアルタイムで通信を行う。PUは、APUによって、データとアプリケーションの処理のスケジューリングと全般的な管理とを行う。APUは並列的かつ独立にこの処理を実行する。DMACは、共用DRAMに格納されているデータとアプリケーションへのアクセス制御をPUとAPUとによって行う。

【0017】このモジュラー構造によれば、ネットワークのメンバーによって用いられるPEの数は、そのメンバーが必要とする処理パワーに基づく。例えば、1台のサーバーは4つのPEを用いることができ、1台のワークステーションは2つのPEを用いることができ、1つのPDAは1つのPEを用いることができる。特定のソフトウェア・セルの処理に割り当てられるPEのAPUの数は、そのセル内のプログラムとデータの複雑さと大きさによって決まる。

【0018】好ましい実施形態では、複数のPEが1つの共用DRAMと関連付けられる。好適には、DRAMは複数のセクションに分割され、これらのセクションの各々は複数のメモリ・バンクに分割される。特に好ましい実施形態では、DRAMは64個のメモリ・バンクを有し、各バンクは1メガバイトの記憶容量を有する。DRAMの各セクションは、好適には、バンク・コントローラによって制御されることが望ましく、PEの各DMACは、好適には、各バンク・コントローラにアクセスすることが望ましい。したがって、この実施形態の各PEのDMACは、共用DRAMの任意の部分へのアクセスが可能となる。

【0019】別の態様では、本発明は、共用DRAMからのAPUのデータの読み出しと、共用DRAMへのデータの書き込みのための同期システムと方法とを提供する。このシステムによって、DRAMを共用している複数のAPUと複数のPEとの間のコンフリクトが防止される。このシステムと方法とによれば、DRAM領域が指定され、複数のフル・エンブディ・ビットが格納され

(9)

特開2002-342165

15

る。これらのブルーエンベティ・ビットの各々は、DRAMの指定領域に対応する。この同期システムはDRAMのハードウェアの中に統合化されるので、ソフトウェアの中で実行されるデータ同期方式の計算上のオーバーヘッドはこのシステムによって防止される。

【0020】また本発明によって、DRAM内にサンドボックスが設けられ、1つのAPUのプログラム処理用データから生じる、別のAPUのプログラム処理用データの破損に対するセキュリティが与えられる。各サンドボックスによって、データの読み出しや書き込みが不可

能となる共用DRAM領域が画定される。

【0021】別の態様では、本発明は、PUがAPUへコマンドを出して、アプリケーションとデータのAPUによる処理を開始するためのシステムと方法とを提供するものである。これらのコマンドは、APU遠隔処理命令(ARPC)と呼ばれ、このコマンドによって、APUがコプロセッサの役割を演じることなく、アプリケーションとデータのAPUによる並列処理のPUによる全般的管理と調整が可能となる。

【0022】他の実施形態では、本発明によって、ストリーミング・データ処理用の専用パイプライン構造を設定するシステムと方法とが提供される。このシステムと方法によれば、PUによってこれらのストリーミング・データの処理を行うために、APUの調整グループと、これらのAPUに関連するメモリサンドボックスの調整グループとが設定される。パイプ・ラインの専用APUとメモリサンドボックスとは、データ処理が行われない時間中もパイプ・ライン専用のままである。言い換えれば、専用APU及びこれらの専用APUと関連するサンドボックスとは、この期間中は予約状態となる。

【0023】他の実施形態では、本発明はタスク処理用の絶対タイマーを提供する。この絶対タイマーは、アプリケーションとデータの処理用としてAPUが使用するクロック周波数に依存しない。アプリケーションは、絶対タイマーによって定義される、タスク用の時間に基づいて書かれる。APUが使用しているクロック周波数が、APUの機能の改善などに起因して増加しても、絶対タイマーによって定義される所定のタスク用の時間はそのまま同じである。この方式によれば、古いAPUにおける遅い処理時間を前提として言われた古いアプリケーションの処理を、これらの新しいAPUでは行わせな

いこととする必要がなく、かつ、新しいバージョンのAPUによる処理時間の向上を実現することが可能になる。

【0024】また本発明は、より処理速度が高速な新しいAPUを、古いAPUにおける遅い処理速度を前提として書かれた古いアプリケーションの処理に用いることを可能にする。他の方式をも提供するものである。この方式では、速度の改善によって生じるAPUの並列処理の調整における問題の処理の間に、これらの古いアプリ

16

ケーションの処理時にAPUが使用している命令(マイクロコード)が分析される。APUによる処理の順番がプログラムが予期する順番どおりに維持されるよう、“オペレーションなし”(“NOOP”)命令が、これらのAPUのいくつかによって実行される命令の中へ挿入される。これらの命令の中へこれらの“NOOP”を挿入することにより、APUによるすべての命令を実行するための正しいタイミングが維持される。

【0025】他の実施形態では、本発明は、光導波路が統合化される集積回路を含むチップ・パッケージを提供するものである。

【0026】

【発明の実施の形態】図1に、本発明によるコンピュータ・システム101のアーキテクチャ全体を示す。

【0027】この図に例示されているように、システム101にはネットワーク104が含まれ、複数のコンピュータとコンピューティング・デバイスがこのネットワークと接続されている。ネットワーク104の例として、LAN、インターネットのようなグローバルネットワーク、あるいは他のコンピュータ・ネットワークが挙げられる。

【0028】ネットワーク104と接続されたコンピュータとコンピューティング・デバイス(ネットワークの“メンバー”)の中には、クライアント側コンピュータ106、サーバーコンピュータ108、個人情報機器(PDA)110、デジタルテレビ(DTV)112およびその他の有線または無線コンピュータとコンピューティング・デバイスなどが含まれる。ネットワーク104のメンバーによって用いられるプロセッサは、同じ共通のコンピューティング・モジュールから構成される。またこれらのプロセッサは、好適には、ISAがすべて同じで、好適には同じ命令セットに従って処理を実行する。個々のプロセッサ内に含まれるモジュールの数は、そのプロセッサが必要とする処理パワーによって決められる。

【0029】例えば、システム101のサーバー108は、クライアント106より多いデータ処理およびアプリケーション処理を実行するので、クライアント106より多いコンピューティング・モジュールを含むことになる。一方、PDA110では最低量の処理しか実行されない。したがって、PDA110には最少の数のコンピューティング・モジュールしか含まれない。DTV112はクライアント106とサーバー108の間の処理レベルを実行する。したがって、DTV112にはクライアント106とサーバー108の間のいくつかのコンピューティング・モジュールが含まれる。以下に説明するように、各コンピューティング・モジュールの中には、処理用コントローラと、ネットワーク104を介して伝送されるデータおよびアプリケーションの並列処理

(10)

特開2002-342165

17

【0030】システム101がこのような均質な構成を有することから、アダプタビリティ、処理速度および処理効率が改善される。システム101の各メンバーが、同じコンピューティング・モジュールのうち1つまたはそれ以上(またはコンピューティング・モジュールの一部)を用いて処理を実行するので、データとアプリケーションの実際の処理をどのコンピュータまたはコンピューティング・デバイスで実行するかは重要ではなくなる。さらに、個々のアプリケーションとデータの処理は、ネットワークのメンバーの間で分担することができる。システム全体を通じて、システム101が処理したデータとアプリケーションを含むセルを一意的に識別することにより、この処理がどこで行われたかにかかわらず、処理を要求したコンピュータまたはコンピューティング・デバイスへその処理結果を伝送することが可能となる。この処理を実行するモジュールが共通の構造と共通のISAとを有するので、プロセッサ間の互換性を達成するためのソフトウェアの追加層の計算上の負担が回避される。このアーキテクチャとプログラミング・モデルによって、リアルタイムのマルチメディア・アプリケーションなどの実行に必要な処理速度が改善される。

【0031】システム101によって改善される処理速度と効率というさらなる利点を利用するために、このシステムによって処理されるデータとアプリケーションとは、一意的に識別される。それぞれフォーマットが同じであるソフトウェア・セル102へとパッケージ化される。各ソフトウェア・セル102は、アプリケーションとデータの双方を含むあるいは含み得る。また各ソフトウェア・セルには、ネットワーク104とシステム101全体の中でセルを識別するためのセル識別子が含まれ、その一例としては、ソフトウェア・セルをグローバルに識別するIDが含まれる。ソフトウェア・セルのこの構造的均一性と、ネットワークの中でのソフトウェア・セルの一意的識別とによって、ネットワークの任意のコンピュータまたはコンピューティング・デバイスでのアプリケーションとデータの処理が改善される。例えば、クライアント106は、ソフトウェア・セル102の作成を行うこともできるが、クライアント106側の処理能力は限られていることから、このソフトウェア・セルをサーバー108へ伝送して処理してもらうこともできる。したがって、ソフトウェア・セルは、ネットワーク104全体を移動してネットワーク上での処理用リソースの可用性に基づく処理を行うことが可能となる。

【0032】また、システム101のプロセッサとソフトウェア・セルが均質な構成を有することで、今日の異質なネットワークの混在という問題の多くを防ぐことができる。例えば任意の命令セットを用いる任意のどのISA上でもアプリケーションの処理を許可しようとする非効率的なプログラミング・モデル(Javaのパーチャル・マシンのような仮想マシンなど)が回避され

18

る。したがって、システム101は、今日のネットワークよりもはるかに効率的、かつ、はるかに効率的に広帯域処理の実現が可能となる。

【0033】ネットワーク104のすべてのメンバーのための基本となる処理用モジュールはプロセッサ・エレメント(PE)である。図2にPEの構造が例示されている。この図に示すように、PE201は、処理ユニット(PU)203、DMAC205、複数の付加処理ユニット(APU)、すなわち、APU207、APU209、APU211、APU213、APU215、APU217、APU219、APU221を具備する。ローカルPEバス223は、APUと、DMAC205と、PU203との間でデータとアプリケーションとを伝送する。ローカルPEバス223は、従来型のアーキテクチャなどを備えていてもよいし、あるいは、パケット交換式ネットワークとして実現されてもよい。パケット交換式ネットワークとして実現される場合、より多くのハードウェアが必要となり、その一方で、利用可能な帯域幅が増加する。

【0034】PE201は、デジタル論理回路を実現する様々な方法を用いて構成可能である。しかし、PE201は、好適には、シリコン基板上の単一の集積回路として構成されることが望ましい。基板用代替材料の中には、ガリウム遷素、ガリウム・アルミニウム遷素、遷素および多種多様のドーパントを用いるその他のいわゆるIII-B化合物が含まれる。またPE201は、超伝導材料(高導単一遷素量子(RSFQ)論理処理など)を用いて実現することもできる。

【0035】PE201は、高帯域メモリ接続部227を介してダイナミック・ランダム・アクセス・メモリ(DRAM)225と密接に関連する。DRAM225はPE201用メイン・メモリとして機能する。DRAM225は好適には、ダイナミック・ランダム・アクセス・メモリであることが望ましいとはいえず、他の手段、例えばスタティック・ランダム・アクセス・メモリ(SRAM)として、磁気ランダム・アクセス・メモリ(MRAM)、光メモリまたはホログラフィ・メモリなどを用いてDRAM225を実現することもできる。DMAC205によって、DRAM225と、PE201のAPUとPUとの間のデータ転送が改善される。以下さらに説明するように、DMAC205によって、各APUに対するDRAM225内の排他的領域が指定されるが、この排他的領域の中へはAPUだけしかデータの書き込みができず、また、APUだけしかこの排他的領域からのデータ読み出しを行うことができない。この排他的領域は“サンドボックス”と呼ばれる。

【0036】PU203は、データとアプリケーションのスタンド・アロン型処理が可能な標準的プロセッサなどであってもよい。作動時に、PUは、APUによって、データとアプリケーションの処理のスケジュール管

(11)

特開2002-342165

19

20

理と全般的な管理とを行う。APUは好適には、単一命令、複数データ(SIMD)プロセッサであることが望ましい。PU203の制御によって、APUは、並列的かつ独立にこれらのデータとアプリケーションの処理を実行する。DMAC205は、共用DRAM225に格納されているデータとアプリケーションへのPU203とAPUによるアクセス制御を行う。PE201は、好適には8個のAPUを含むことが望ましいとはいえず、必要とする処理パワーに応じて、PE内でこの数より多少上下する個数のAPUを用いてもよい。また、PE201

のようないくつかのPEを結合(まとめてパッケージ化)して処理パワーの改善を図ることもできる。
【0037】例えば、図3に示すように、1以上のチップ・パッケージなどの中に4つのPEをパッケージ化(まとめて結合)してネットワーク104のメンバー用の単一プロセッサを形成してもよい。この構成は広帯域エンジン(BE)と呼ばれる。図3に示すように、BE301には4つのPE(PE303、PE305、PE307、PE309)が含まれる。これらのPE間の通信はBEバス311を介して行われる。広帯域メモリ接続部313によって共用DRAM315とこれらのPE間の通信が行われる。BEバス311の代わりに、BE301のPE間の通信は、DRAM315とこのメモリ接続部とを介して行うことができる。

【0038】入力/出力(I/O)インターフェース317と外部バス319とは、広帯域エンジン301とネットワーク104のその他のメンバー間で通信を行う。BE301の各PEは、PEのAPUによって行われるアプリケーションとデータの並列的かつ独立した処理と同様の並列的かつ独立した方法で、データとアプリケーションの処理を実行する。

【0039】図4はAPUの構造を例示する図である。APU402には、ローカル・メモリ406、レジスタ410、4つの浮動小数点演算ユニット412および4つの整数演算ユニット414が含まれる。しかし、ここでもまた、必要とする処理パワーに応じて、4個より多少上下する個数の浮動小数点演算ユニット412と整数演算ユニット414を用いてもよい。1つの好ましい実施形態では、ローカル・メモリ406には128キロバイトの記憶容量が含まれ、レジスタ410の容量は128×128ビットである。浮動小数点演算ユニット412は、毎秒320億浮動小数点演算(32GLOPS)の速度で好適に作動し、整数演算ユニット414は、毎秒320億回の演算速度(32GOP)で好適に作動する。

【0040】ローカル・メモリ406はキャッシュ・メモリではない。ローカル・メモリ406は、好適にはSRAMとして構成されることが望ましい。APUに対するキャッシュ・コヒーレンシー、つまりキャッシュの整合性のサポートは不要である。PUでは、当該PUで開始されるダイレクト・メモリー・アクセス(DMA)をサ

ポートするためにキャッシュの整合性が要求される場合もある。しかし、APUによって開始されるDMAに対する、あるいは、外部装置からのおよび外部装置へのアクセスに対するキャッシュの整合性のサポートは不要である。

【0041】APU402にはさらに、APUへおよびAPUからアプリケーションとデータとを伝送するためのバス404が含まれる。1つの好ましい実施形態ではこのバスは1024ビットの幅を持つ。APU402にはさらに内部バス408、420、418が含まれる。1つの好ましい実施形態では、バス408は256ビットの幅を持ち、ローカル・メモリ406とレジスタ410間で通信を行う。バス420と418とは、それぞれ、レジスタ410と浮動小数点演算ユニット412との間、および、レジスタ410と整数演算ユニット414間で通信を行う。ある好ましい実施形態では、レジスタ410から浮動小数点演算ユニット412または整数演算ユニット414へのバス418と420の幅は、384ビットであり、浮動小数点演算ユニット412または整数演算ユニット414からレジスタ410へのバス418と420の幅は128ビットである。浮動小数点演算ユニット412または整数演算ユニット414からレジスタ410への幅より広い、レジスタ410から浮動小数点演算ユニットまたは整数演算ユニットへの上記バスの広い幅によって、レジスタ410からのより広いデータ・フローが処理中に許容される。最大3ワードが各計算には必要となる。しかし、各計算の結果は、一般に、ただ1ワードだけである。

【0042】図5～10は、ネットワーク104のメンバーのプロセッサのモジュラー構造をさらに例示する図である。例えば、図5に示すように、1つのプロセッサには単一のPE502を含むことができる。上述のように、このPEには、一般に、PU、DMACおよび8個のAPUが含まれる。各APUにはローカル・ストレージ(LS)が含まれる。一方、プロセッサは、ビジュアル・イザ(VS)505の構造を有する場合もある。図5に示すように、VS505はPU512、DMAC514および4つのAPU(APU516、APU518、APU520、APU522)を有する。PEのその他の4つのAPUによって通常占有されるチップ・パッケージ内のスペースは、この場合、ピクセル・エンジン508、画像用キャッシュ510およびブラウン管コントローラ(CRTC)504によって占有される。PE502またはVS505に求められる通信速度に応じて、チップ・パッケージの中に光インターフェース506が含まれる場合もある。

【0043】この標準化されたモジュラー構造を用いて、多数の他のプロセッサの変更例を容易にかつ効率的に構成することが可能となる。例えば、図6に示すプロセッサは、2つのチップ・パッケージ(BE)を備えるチ

(12)

特開2002-342165

21

ップ・パッケージ602と、4つのVSを含むチップ・パッケージ604)を有する。入出力部(I/O)606によって、チップ・パッケージ602のBEとネットワーク104との間にインターフェースが設けられる。バス608はチップ・パッケージ602とチップ・パッケージ604との間の通信を行う。入出力プロセッサ(IOP)610によってデータ・フローが制御され、I/O606へのまたはI/O606からの入出力が行われる。I/O606はASIC(Application Specific Integrated Circuit)として製造が可能である。VSから10

【0044】図7は、ネットワーク104のその他のメンバーへ超高速通信を行う2つの光インターフェース704と706とを備えたBE702用のチップ・パッケージ(またはローカルに接続された他のチップ・パッケージ)を例示する。BE702は、ネットワーク104上でサーバーなどとして機能することができる。

【0045】図8のチップ・パッケージは、2つのPE802及び804および2つのVS806及び808を有する。I/O810は、チップ・パッケージとネットワーク104との間にインターフェースを与える。チップ・パッケージからの出力はビデオ信号である。この構成は画像処理用ワークステーションなどとして機能することができる。

【0046】図9はさらに別の構成を例示する。この構成は、図8に例示されている構成の処理パワーの1/2を含む。2つのPEの代わりに1つのPE902が設けられ、2つのVSの代わりに1つのVS904が設けられる。I/O906は、図8に例示されているI/Oの帯域幅の1/2の帯域幅を有する。またこのようなプロセッサは、画像処理用ワークステーションとして機能することができる。

【0047】最後の構成が図10に図示されている。このプロセッサは、単一のVS1002とI/O1004だけから構成される。この構成はPDAなどとして機能することができる。

【0048】図11は、ネットワーク104のプロセッサのチップ・パッケージの中への光インターフェースの統合を例示する図である。これらの光インターフェースによって、光信号は電気信号に変換され、電気信号は光信号に変換される。また、これらの光インターフェースは、ガリウム砒素、アルミニウム・ガリウム砒素、ゲルマニウムその他の元素や化合物などを含む様々な材料から構成することができる。この図に示すように、光インターフェース1104と1106とはBE1102のチップ・パッケージの上に組み立てられる。BEバス1108はBE1102のPE、すなわち、PE1110、PE1112、PE1114、PE1116およびこれらの光インターフェースとの間での通信を行う。光インターフェース1104には2つのポート(ポート111

22

8とポート1120)が含まれ、また光インターフェース1106には2つのポート(ポート1122とポート1124)が含まれる。ポート1118、1120、1122、1124は、光導波路1126、1128、1130、1132とそれぞれ接続される。光信号は、光インターフェース1104と1106のポートを介して、これらの光導波路の中を通り、BE1102へおよびBE1102から伝送される。

【0049】このような光導波路と各BEの4つの光ポートとを用いて様々な構成において複数のBEをまとめて接続してもよい。例えば、図12に示すように、このような光ポートを介して2つまたはそれ以上のBE(BE1152、BE1154、BE1156など)を直列に接続することができる。この例では、BE1152の光インターフェース1166は、その光ポートを介しBE1154の光インターフェース1160の光ポートと接続される。同様に、BE1154の光インターフェース1162の光ポートは、BE1156の光インターフェース1164の光ポートと接続される。

【0050】図13にマトリクス構成が例示される。この構成では、各BEの光インターフェースは2つの他のBEと接続される。この図に示すように、BE1172の光インターフェース1188の光ポートの中の1つが、BE1176の光インターフェース1182の光ポートと接続される。光インターフェース1188のもう一方の光ポートは、BE1178の光インターフェース1184の光ポートと接続される。同様に、BE1174の光インターフェース1190の1つの光ポートはBE1178の光インターフェース1184のもう一方の光ポートと接続される。光インターフェース1190のもう一方の光ポートは、BE1180の光インターフェース1186の光ポートと接続される。このマトリクス構成は他のBEに対しても同様に拡張することができる。

【0051】シリアル構成かマトリクス構成のいずれかを用いて、任意の所望のサイズとパワーから成るネットワーク104用プロセッサの構成が可能となる。言うまでもなく、BEの光インターフェースに対して、または、BEよりPE数の少ないプロセッサに対して追加ポートを加えて、他の構成を形成してもよい。

【0052】図14はBEのDRAMに対する制御システムと構造を例示する図である。同様の制御システムと構造が、別のサイズを持ち、多少異なる数のPEを含むプロセッサの中で用いられる。この図に示すように、クロスバ交換機によって、BE1201を備える4つのPEからなる各DMAC1210が8つのバンク・コントロール1206と接続される。各バンク・コントロール1206によって、DRAM1204の8つのバンク1208(4つだけしか図示されていない)が制御される。したがって、DRAM1204は、合計64のバンクを

(13)

特開2002-342165

23

24

具備することになる。好ましい実施形態では、DRAM 1204は64メガバイトの容量を持ち、各バンクは1メガバイトの容量を持っている。各バンク内の最小のアドレス指定可能単位は、この好ましい実施形態では1024ビットのブロックである。

【0053】BE1201にはスイッチ・ユニット1212も含まれる。スイッチ・ユニット1212によって、BE1201と密接に接続されているBEの他のAPUのDRAM1204へのアクセスが可能となる。したがって、第2のBEを第1のBEと密接に接続することが可能となり、さらに、各BEの各APUは、APUが通常アクセス可能なメモリ・ロケーションの数の2倍のアドレス指定を行うことが可能となる。スイッチ・ユニット1212のようなスイッチ・ユニットを介して、第1のBEのDRAMから第2のBEのDRAMへのデータの直接読み出し、または、第2のBEのDRAMから第1のBEのDRAMへのデータの直接書き込みを行うことが可能となる。

【0054】例えば、図15に示すように、このような書き込みを行うために、第1のBEのAPU(BE1222のAPU1220など)によって、第2のBEのDRAM(通常の場合のようなBE1222のDRAM1224ではなく、BE1226のDRAM1228など)のメモリ・ロケーションへの書き込みコマンドが出される。BE1222のDMAC1230は、クロスバ交換機1221を介して、バンク・コントロール1234へ書き込みコマンドを送り、バンク・コントロール1234は、バンク・コントロール1234と接続された外部ポート1232へコマンドを伝送する。BE1226のDMAC1238は書き込みコマンドを受け取り、BE1226のスイッチ・ユニット1240はこのコマンドを転送する。スイッチ・ユニット1240は書き込みコマンドの中に含まれるDRAMアドレスを識別し、BE1226のバンク・コントロール1242を介して、DRAM1228のバンク1244へ、DRAMアドレス内に格納するデータを送る。したがって、スイッチ・ユニット1240によって、DRAM1224とDRAM1228の双方は、BE1222のAPU用の単一メモリ空間として機能することが可能になる。

【0055】図16はDRAMの64個のバンク構成を図示する。これらのバンクは、8つの行(1302、1304、1306、1308、1310、1312、1314、1316)と8つの列(1320、1322、1324、1326、1328、1330、1332、1334)とで構成されている。各行は1つのバンク・コントローラによって制御される。したがって、各バンク・コントローラは8メガバイトのメモリを制御する。

【0056】図17と18は、最小のアドレス指定可能格納単位(1024ビットのブロックなど)でのDRAMの格納とアクセスを行うための異なる構成を例示す

る。図17で、DMAC1402は単一のバンク1404の中に8つの1024ビット・ブロック1406を格納する。図18では、DMAC1412によって、1024ビットを含むデータ・ブロックの読み出しと書き込みが行われるものの、これらのブロックは、2つのバンク(バンク1414とバンク1416)の間で分配される。したがって、これらのバンクの各々には16個のデータ・ブロックが含まれ、データの各ブロックには512ビットが含まれる。この分配によって、DRAMのアクセスをさらに高速なものに改善することが可能となり、ある種のアプリケーションの処理に役立つ。

【0057】図19はPE内のDMAC1506のアーキテクチャを例示する。この図に例示されているように、各APU1502がDMAC1506の構造上のノード1504へ直接アクセスを行うように、DMAC1506を含む構造上のハードウェアは全てのPEを通じて配設される。各ノードは、ノードが直接アクセスを行う対象のAPUによるメモリ・アクセスに適した論理処理を実行する。

【0058】図20はDMACの他の実施形態、すなわち、非分配型アーキテクチャを図示する。この場合、DMAC1606の構造上のハードウェアは集中型である。APU1602とPU1604は、ローカルPEバス1607を介してDMAC1606を用いて通信を行う。DMAC1606はクロスバ・スイッチを介してバス1608と接続される。バス1608はDRAM1610と接続されている。

【0059】上述のように1つのPEの複数のAPUのすべては、独立に、共用DRAM内のデータへのアクセスが可能である。その結果、第1のAPUがあるデータをそのローカル・ストレージで処理しているときに、第2のAPUがこれらのデータを要求する場合もある。その時点で共用DRAMから第2のAPUへ当該データが出力された場合、データの値を変化させ得る第1のAPUの進行中の処理に起因して、そのデータが無効になる場合がある。したがって、その時点で第2のプロセッサが共用DRAMからデータを受け取った場合、第2のプロセッサでエラー結果が生じるおそれがある。例えば、このようなデータとしては、グローバル変数用の具体的な値が挙げられる。第1のプロセッサがその処理中その値を変えた場合、第2のプロセッサはもう使用されていない値を受け取ることになる。したがって、共用DRAMの範囲内でメモリ・ロケーションからのおよびメモリ・ロケーションへのAPUによるデータの読み出しと書き込みを同期させる何らかの方法が必要となる。この方式では、別のAPUがそのローカル・ストレージで現在働かしている対象データであって、したがって最新のものではないデータのメモリ・ロケーションからの読み出しと、最新のデータを格納するメモリ・ロケーションの中へのデータの書き込みと、を行わないようにする必

(14)

特開2002-342165

25

26

要がある。

【0060】これらの問題を解決するために、DRAMの各アドレス指定が可能なメモリ・ロケーションに対して、そのメモリ・ロケーションの中に格納されているデータに関連する状態情報を格納するために、DRAMの中でメモリの追加セグメントの割り振りが行われる。この状態情報の中には、フル／エンプティ(F/E)ビットと、メモリ・ロケーションからデータを要求するAPUの識別子(APU ID)と、要求されたデータを読み出す読み出し先となるAPUのローカル・ストレージのアドレス(LSアドレス)とが含まれる。DRAMのアドレス指定が可能なメモリ・ロケーションは任意のサイズとすることができる。ある好ましい実施形態ではこのサイズは1024ビットである。

【0061】F/Eビットの1への設定は、メモリ・ロケーションに格納されているデータが最新のものであることを示す。一方、F/Eビットの0への設定は、関連するメモリ・ロケーションに格納されたデータが最新のものではないことを示す。このビットが0に設定されているとき、APUがそのデータを要求しても、APUによってそのデータの即時読み出しは妨げられる。この場合、そのデータを要求しているAPUを識別するAPU IDと、データが最新のものになっているとき、そのデータを読み出す読み出し先となるこのAPUのローカル・ストレージ内のメモリ・ロケーションを識別するLSアドレスとが、追加メモリ・セグメントの中へ入力される。

【0062】また追加メモリ・セグメントは、APUのローカル・ストレージ内の各メモリ・ロケーションに対して割り振られる。この追加メモリ・セグメントは、“ビジー・ビット”と呼ばれる1ビットを格納する。このビジー・ビットは、DRAMから検索される固有データの格納用として関連するLSメモリ・ロケーションの予約を行うために使用される。ローカル・ストレージ内の特定のメモリ・ロケーションに対してビジー・ビットが1に設定されている場合、これらの固有データの書き込み用としてのみAPUはこのメモリ・ロケーションを使用することができる。一方、ビジー・ビットが、ローカル・ストレージ内の特定のメモリ・ロケーションに対して0に設定されている場合、APUは、任意のデータの書き込み用としてこのメモリ・ロケーションを使用することができる。

【0063】F/Eビット、APU ID、LSアドレスおよびビジー・ビットが、PEの共用DRAMからの、および、PEの共用DRAMへのデータの読み出しと書き込みを同期させるために使用される方法を示す例が図21-35に例示されている。

【0064】図21に示すように、1以上のPE(PE 1720など)がDRAM1702を使用する。PE 1720にはAPU1722とAPU1740とが含まれ

る。APU1722には制御論理回路1724が含まれ、APU1740には制御論理回路1742が含まれる。APU1722にはローカル・ストレージ1726も含まれる。このローカル・ストレージには複数のアドレス可能なメモリ・ロケーション1728が含まれる。APU1740にはローカル・ストレージ1744が含まれ、このローカル・ストレージにも複数のアドレス可能なメモリ・ロケーション1746が含まれる。これらのアドレス可能なメモリ・ロケーションのすべては好適にはサイズが1024ビットであることが望ましい。

【0065】メモリの追加セグメントは各LSのアドレス可能なメモリ・ロケーションと関連付けられる。例えば、メモリ・セグメント1729と1734とはそれぞれ、ローカルなメモリ・ロケーション1731と1732とに関連付けられ、メモリ・セグメント1752はローカルなメモリ・ロケーション1750と関連付けられる。上述のような“ビジー・ビット”はこれらの追加メモリ・セグメントの各々の中に格納される。ローカルなメモリ・ロケーション1732は、このメモリ・ロケーションがデータを含むことを示すいくつかの×印を用いて示されている。

【0066】DRAM1702には、メモリ・ロケーション1706と1708とを含む複数のアドレス可能なメモリ・ロケーション1704が含まれる。これらのメモリ・ロケーションは、好適にはサイズが1024ビットであることが望ましい。メモリの追加セグメントはまたこれらのメモリ・ロケーションの各々とも関連付けられる。例えば、追加メモリ・セグメント1760はメモリ・ロケーション1706と関連し、追加メモリ・セグメント1762はメモリ・ロケーション1708と関連付けられる。各メモリ・ロケーションに格納されるデータに関連する状態情報は、メモリ・ロケーションと関連付けられたメモリ・セグメントに格納される。この状態情報の中には、上述のように、F/Eビット、APU IDおよびLSアドレスが含まれる。例えば、メモリ・ロケーション1708については、この状態情報にはF/Eビット1712、APU ID 1714およびLSアドレス1716が含まれる。

【0067】この状態情報とビジー・ビットとを用いて、PEのAPU、または1グループのPE間での、共用DRAMからの、および、同期した共用DRAMからの読み出しと、同期した共用DRAMへのデータの書き込みを行うことができる。

【0068】図22はAPU1722のLSメモリ・ロケーション1732から、DRAM1702のメモリ・ロケーション1708へのデータの同期書き込みの開始を例示する図である。APU1722の制御論理回路1724によってこれらのデータの同期書き込みが開始される。メモリ・ロケーション1708がエンプティであるため、F/Eビット1712は0に設定される。その

(15)

特開2002-342165

27

結果、メモリ・ロケーション1708の中へLSメモリ・ロケーション1732内のデータを書き込むことが可能となる。一方、このビットが1に設定されていて、メモリ・ロケーション1708がフル状態であり、最新の有効データを含むことが示されている場合、制御回路1722はエラー・メッセージを受け取ることになり、このメモリ・ロケーションへのデータの書き込みは禁止される。

【0069】メモリ・ロケーション1708への成功したデータの同期書き込みの結果が図23に示されている。この書き込まれたデータはメモリ・ロケーション1708の中に格納され、F/Eビット1712は1に設定される。この設定によって、メモリ・ロケーション1708がフル状態であること、および、このメモリ・ロケーションの中のデータが最新の有効データであることが示される。

【0070】図24は、DRAM1702のメモリ・ロケーション1708からローカル・ストレージ1744のLSメモリ・ロケーション1750へのデータの同期読み出しの開始を例示する図である。この読み出しを開始するために、LSメモリ・ロケーション1750のメモリ・セグメント1752の中のビジー・ビットが1に設定されて、上記データ用としてこのメモリ・ロケーションが予約される。このビジー・ビットを1に設定することによって、APU1740がこのメモリ・ロケーションに他のデータを格納することはなくなっている。

【0071】図25に示すように、制御論理回路1742は次にDRAM1702のメモリ・ロケーション1708に対して同期読取りコマンドを出す。このメモリ・ロケーションと関連付けられるF/Eビット1712は1に設定されているので、メモリ・ロケーション1708の中に格納されたデータは最新の、有効データであると見なされる。その結果、メモリ・ロケーション1708からLSメモリ・ロケーション1750へのデータ転送の準備の際に、F/Eビット1712は0に設定される。この設定が図26に示されている。このビットを0に設定されているということは、これらのデータの読み出しの後に、メモリ・ロケーション1708のデータは無効になることを示す。

【0072】図27に示すように、メモリ・ロケーション1708内のデータは、次に、メモリ・ロケーション1708からLSメモリ・ロケーション1750へ読み出される。図28は最終状態を示す図である。メモリ・ロケーション1708のデータのコピーはLSメモリ・ロケーション1750に格納される。F/Eビット1712は0に設定され、メモリ・ロケーション1708のデータが無効であることが示される。この無効は、APU1740によって行われた上記データの変更の結果である。メモリ・セグメント1752内のビジー・ビットもまた0に設定される。この設定によって、APU17

28

40がLSメモリ・ロケーション1750を任意の目的に利用できること、すなわち、このLSメモリ・ロケーションがもはや固有データの受信を待機している予約状態ではないことが示される。したがって、任意の目的のためにAPU1740によるLSメモリ・ロケーション1750へのアクセスが可能となる。

【0073】図29～図35には、DRAM1702のメモリ・ロケーション用のF/Eビットが、0に設定されていて、このメモリ・ロケーションのデータが最新のものでもなく有効なものでもないことが示されている場合の、DRAM1702(メモリ・ロケーション1708など)のメモリ・ロケーションから、APUのローカル・ストレージ(ローカル・ストレージ1744のLSメモリ・ロケーション1752など)のLSメモリ・ロケーションへのデータの同期読み出しが例示されている。図29に示すように、この転送を開始するために、LSメモリ・ロケーション1750のメモリ・セグメント1752内のビジー・ビットは1に設定され、このデータ転送用としてこのLSメモリ・ロケーションが予約される。図30に示すように、制御論理回路1742は、次に、DRAM1702のメモリ・ロケーション1708に対して同期読取りコマンドを出す。このメモリ・ロケーションと関連付けられたF/Eビット(F/Eビット1712)は0に設定されているので、メモリ・ロケーション1708に格納されているデータは無効である。その結果、信号は制御論理回路1742へ伝送され、このメモリ・ロケーションからのデータの即時読み出しが阻止される。

【0074】図31に示すように、APU ID1714とこの読取りコマンド用のLSアドレス1716とはメモリ・セグメント1762の中へ書き込まれる。この場合、APU1740用のAPU IDと、LSメモリ・ロケーション1750用のLSメモリ・ロケーションとはメモリ・セグメント1762の中へ書き込まれる。したがって、メモリ・ロケーション1708の範囲内のデータが最新のものになっているとき、このAPU IDとLSメモリ・ロケーションは、最新のデータを伝送する伝送先のメモリ・ロケーションを決定するために使用される。

【0075】メモリ・ロケーション1708内のデータは、APUがこのメモリ・ロケーションの中へデータを書き込むと、有効で最新のデータとなる。APU1722のメモリ・ロケーション1732などからメモリ・ロケーション1708の中へのデータの同期書き込みが図29に例示されている。このメモリ・ロケーション用のF/Eビット1712が0に設定されているため、これらのデータのこの同期書き込みは許される。

【0076】図33に示すように、この書き込み後、メモリ・ロケーション1708の中のデータは最新の有効データになる。したがって、メモリ・セグメント176

(16)

特開2002-342165

29

30

2から得られるAPUID1714とLSアドレス1716とは、メモリ・セグメント1762から即座に読み出され、次いでこの情報はこのセグメントから削除される。メモリ・ロケーション1708の中のデータの即時読み出しを予期して、F/Eビット1712もまた0に設定される。図34に示すように、APUID1714とLSアドレス1716とを読み出すと、APU1740のLSメモリ・ロケーション1750へメモリ・ロケーション1708内の有効データを読み出すためにこの情報は直ちに使用される。最終状態が図35に図示されている。この図は、メモリ・ロケーション1708からメモリ・ロケーション1750へコピーされた有効データと、0に設定されたメモリ・セグメント1752内のビジー・ビットと、0に設定されたメモリ・セグメント1762内のF/Eビット1712とを図示する。このビジー・ビットの0への設定によって、任意の目的のためにAPU1740がLSメモリ・ロケーション1750のアクセスを行うことが可能になる。このF/Eビットの0への設定によって、メモリ・ロケーション1708内のデータがもはや最新のものでもなく、有効なものでもないことが示される。

【0077】図36は、上述のオペレーションと、DRAMのメモリ・ロケーションの様々な状態とを要約する図であり、この状態は、F/Eビットの状態と、APUIDと、メモリ・ロケーションに対応するメモリ・セグメントの中に格納されたLSアドレスとに基づく。このメモリ・ロケーションは、3つの状態を持つことが可能である。これらの3つの状態として、F/Eビットが0に設定され、APUIDまたはLSアドレスに対して情報が提供されないエンプティ状態1880と、F/Eビットが1に設定され、APUIDまたはLSアドレスに対して情報が提供されないフル状態1882と、F/Eビットが0に設定され、APUIDとLSアドレスに対して情報が提供されるブロッキング状態1884とがある。

【0078】この図に示すように、エンプティ状態1880では、同期書き込みオペレーションが許され、フル状態1882への遷移という結果が得られる。しかし、メモリ・ロケーションがエンプティ状態であるときはメモリ・ロケーション内のデータが最新のものではないので、同期読み出しオペレーションに対しては、ブロッキング状態1884へ遷移するという結果となる。

【0079】フル状態1882では、同期読み出しオペレーションが許され、エンプティ状態1880への遷移という結果が得られる。一方、有効データの上書きを避けるために、フル状態1882の同期書き込みオペレーションは禁止される。このような書き込みオペレーションがこの状態で試みられる場合、状態の変化は生じず、エラー・メッセージがAPUの対応する制御論理回路へ伝送される。

【0080】ブロッキング状態1884では、メモリ・ロケーションの中へのデータの同期書き込みが許され、エンプティ状態1880への遷移という結果が得られる。一方、ブロッキング状態1884での同期読み出しオペレーションは禁止される。このブロッキング状態を生じさせることとなった前回同期読み出しオペレーションとのコンフリクトを阻止するためである。同期読み出しオペレーションが、ブロッキング状態1884で試みられた場合、状態変化は生じないでAPUの対応する制御論理回路へエラー・メッセージが伝送される。

【0081】共用DRAMからのデータの同期読み出しと、共用DRAMへのデータの同期書き込みを行う上述の方式は、外部装置からのデータ読み出しと外部装置へのデータ書き込み用プロセッサとして通常専用の計算用リソースを取り除くためにも利用が可能である。この入出力(I/O)機能はPUによって行うこともできる。しかし、この同期方式の変更を利用して、適切なプログラムを実行するAPUがこの機能を実行してもよい。例えば、この方式を利用して、外部装置によって開始された、I/Oインターフェースからのデータ伝送を求める割込み要求を受け取るPUは、このAPUにこの要求の処理を委任してもよい。次いで、APUはI/Oインターフェースに対して同期書き込みコマンドを出す。今度はこのインターフェースによって、現在DRAMの中へデータを書き込むことができる旨の信号が外部装置へ送られる。次にAPUはDRAMに対して同期読取りコマンドを出し、DRAMの関連するメモリ空間をブロッキング状態に設定する。APUはまた、データを受け取る必要があるAPUのローカル・ストレージのメモリ・ロケーションに対してビジー・ビットを1に設定する。ブロッキング状態では、DRAMの関連するメモリ空間と関連付けられた追加メモリ・セグメントの中に、APUのIDとAPUのローカル・ストレージの関連するメモリ・ロケーションのアドレスが含まれる。次に外部装置は同期書き込みコマンドを出し、DRAMの関連するメモリ空間へデータが直接書き込まれる。このメモリ空間はブロッキング状態にあるので、データは、このスペースの中から、追加メモリ・セグメントの中で識別されたAPUのローカル・ストレージのメモリ・ロケーションの中へ直ちに読み出される。次いで、これらのメモリ・ロケーション用のビジー・ビットは0に設定される。外部装置がデータの書き込みを完了したとき、APUは、伝送が完了した旨を示す信号をPUへ出す。

【0082】したがって、この方式を用いて、PUに対する最小の計算上の負荷で、外部装置からのデータ転送処理を行うことができる。しかし、この機能を委任されたAPUはPUに対して割込み要求を出せることが望ましく、外部装置がDRAMに対して直接アクセスを行うことが望ましい。

【0083】各PEのDRAMには複数の“サンドボッ

(17)

特開2002-342165

31

クス”が含まれる。サンドボックスによって共用DRAM領域が固定され、この領域を越えて、特定のAPUまたは1組のAPUがデータの読み出しや書き込みを行うことはできない。これらのサンドボックスによって、1つのAPUが処理するデータに起因する、別のAPUによって処理されるデータの破損に対するセキュリティが与えられる。またこれらのサンドボックスによって、ソフトウェア・セルが全DRAMの中でデータの破損を生じる可能性なく、ネットワーク104から特定のサンドボックスの中へソフトウェア・セルのダウンロードを行うことが許される。本発明では、サンドボックスは、DRAMとDMACとから成るハードウェアの中に設けられる。ソフトウェアの代わりに、このハードウェア内にこれらのサンドボックスを設けることにより、速度とセキュリティという利点を得られる。

【0084】PEのPUはAPUへ割り当てられるサンドボックスの制御を行う。PUは、オペレーティング・システムのような信頼のおけるプログラムだけしか通常動作させないので、本方式によってセキュリティが危険にさらされることはない。本方式に従って、PUはキー管理テーブルの構築と維持とを行う。図37にこのキー管理テーブルが例示されている。この図に示すように、キー管理テーブル1902内の各エントリには、APU用の識別子(ID)1904と、そのAPU用のAPUキー1906と、キー・マスク1908とが含まれる。このキー・マスクの用途について以下説明する。キー管理テーブル1902は、スタティック・ランダム・アクセス・メモリ(SRAM)のような比較的高速のメモリに好適に格納され、DMACと関連付けられる。キー管理テーブル1902へのエントリはPUによって制御される。APUが、DRAMの特定の格納位置(ストレージロケーション)へのデータの書き込みあるいはDRAMの特定の格納位置からのデータの読み出しを要求すると、DMACは、その格納位置と関連付けられたメモリ・アクセス・キーに対して、キー管理テーブル1902内のそのAPUへ割り当てられたAPUキー1906の評価を行う。

【0085】図38に示すように、DRAM2002の各アドレス可能な格納位置2006に対して専用メモリ・セグメント2010が割り当てられる。この格納位置用のメモリ・アクセス・キー2012はこの専用メモリ・セグメントの中に格納される。上述のように、やはり各アドレス可能な格納位置2006と関連付けられたさらなる追加専用メモリ・セグメント2008によって、格納位置へのデータ書き込みと、格納位置からのデータの読み出しを行うための同期情報が格納される。

【0086】作動時に、APUはDMACへDMAコマンドを出す。このコマンドには、DRAM2002の格納位置2006のアドレスが含まれる。このコマンドを実行する前に、DMACは、キー管理テーブル1902

32

におけるAPUのID1904を用いて要求を行っているAPUのキー1906を調べる。次いで、DMACは、APUがアクセスを求める対象先であるDRAMの格納位置と関連付けられた専用メモリ・セグメント2010内に格納されているメモリ・アクセス・キー2012と、要求を行っているAPUのAPUキー1906との比較を行う。2つのキーが一致しない場合、DMAコマンドは実行されない。一方、2つのキーが一致した場合、DMAコマンドは進行し、要求されたメモリ・アクセスが実行される。

【0087】図39に他の実施形態の一例を示す。この例では、PUはメモリ・アクセス管理テーブル2102の格納も行う。メモリ・アクセス管理テーブル2102にはDRAM内にある各サンドボックス用のエントリが含まれる。図39の特定の例では、DRAMには64個のサンドボックスが含まれる。メモリ・アクセス管理テーブル2102内の各エントリには、サンドボックス用識別子(ID)2104と、ベース・メモリ・アドレス2106と、サンドボックス・サイズ2108と、メモリ・アクセス・キー2110と、アクセス・キー・マスク2110とが含まれる。ベース・メモリ・アドレス2106によって、DRAM内にアドレスが設けられ、このアドレスによって特定のメモリ・サンドボックスの最初の部分が示される。サンドボックス・サイズ2108によってサンドボックスのサイズが与えられ、したがって、このサイズによって特定のサンドボックスのエンドポイントが与えられる。

【0088】図40は、キー管理テーブル1902とメモリ・アクセス管理テーブル2102とを用いてDMAコマンドを実行するためのステップを示すフロー・チャートである。ステップ2202では、APUによって、サンドボックス内の特定の1つあるいは複数のメモリ・ロケーションに対するアクセス用DMAコマンドがDMACへ出される。このコマンドには、アクセス要求を行う対象先である特定のサンドボックスの識別を行うサンドボックスID2104が含まれる。ステップ2204では、DMACは、APUのID1904を利用して、キー管理テーブル1902内の要求を行っているAPUのキー1906を調べる。ステップ2206で、DMACは、メモリ・アクセス管理テーブル2102で、サンドボックスと関連付けられたメモリ・アクセス・キー2110を調べるコマンドで、サンドボックスID2104を利用する。ステップ2208で、DMACは、要求を行っているAPUへ割り当てられているAPUキー1906をサンドボックスと関連付けられたアクセス・キー2110と比較する。ステップ2210で、この2つのキーが一致するかどうかの決定が行われる。この2つのキーが一致しない場合、処理はステップ2212へ移行し、そこでDMAコマンドは先へ進まず、要求を行っているAPUとPUのいずれかまたはその双方へエラー

(18)

特開2002-342165

33

34

・メッセージが送信される。一方、ステップ2210で、2つのキーの一致が得られた場合、処理はステップ2214へ進み、そこでDMAはDMAコマンドを実行する。

【0089】APUキー用およびメモリ・アクセス・キー用のキー・マスクによってこのシステムに大きな柔軟性が与えられる。キー用のキー・マスクによって、マスクされたビットはワイルド・カードに変換される。例えば、APUキー1906と関連付けられたキー・マスク1908が、キー・マスク1908内のこれらのビットを1に設定することなどにより、その最後の2ビットが“マスク”に設定されている場合、APUキーは1または0のいずれかになることができ、そのままメモリ・アクセス・キーに一致することになる。例えば、APUキーが1010であるとする。通常、このAPUキーによって1010のアクセス・キーを持つサンドボックスへのアクセスだけが可能になる。しかし、このAPUキー用のAPUキー・マスクが0001に設定されている場合、このAPUキーを用いて1010または1011のいずれかのアクセス・キーを持つサンドボックスへのアクセスを行うことが可能となる。同様に、1010または1011のいずれかのAPUキーを持つAPUによって、0001に設定されたマスクを持つアクセス・キー1010のアクセスを行うことが可能である。APUキー・マスクとメモリ・キー・マスクの双方を同時に使用することができるので、多数のバリエーションのサンドボックスに対するAPUによるアクセシビリティの設定が可能となる。

【0090】また本発明はシステム101のプロセッサ用の新しいプログラミング・モデルも提供するものである。このプログラミング・モデルではソフトウェア・セル102が用いられる。ネットワーク104上の任意のプロセッサへ処理用としてこれらのセルの伝送を行うことが可能である。またこの新しいプログラミング・モデルでは、システム101のユニークなモジュラー形アーキテクチャと、システム101のプロセッサとが利用される。

【0091】ソフトウェア・セルはAPUのローカル・ストレージからAPUによって直接処理される。APUは、DRAM内のいずれのデータまたはプログラムに対しても直接働きかけることは行わない。DRAM内のデータとプログラムは、APUがこれらのデータとプログラムの処理を行う前に、APUのローカル・ストレージの中に読み込まれる。したがって、APUのローカル・ストレージには、プログラム・カウンタと、スタックと、これらのプログラムを実行するための他のソフトウェア・エレメントとが含まれることになる。PUは、DMAに対してDMAコマンドを出すことによりAPUの制御を行う。

【0092】ソフトウェア・セル102の構造が図41

に例示されている。この図に示すように、ソフトウェア・セル2302などのソフトウェア・セルの中には、ルート選定情報セクション2304と本体部分2306とが含まれる。ルート選定情報セクション2304に含まれる情報は、ネットワーク104のプロトコルに依って決められる。ルート選定情報セクション2304の中には、ヘッダ2308、宛先ID2310、ソースID2312および応答ID2314が含まれる。宛先IDにはネットワーク・アドレスが含まれる。TCP/IPプロトコルの下で、例えば、ネットワーク・アドレスはインターネット・プロトコル(IP)アドレスである。さらに宛先ID2310には、処理のためにセルを送送すべき伝送先のPE及びAPUの識別子が含まれる。ソースID2314にはネットワーク・アドレスが含まれ、このソースIDによってPEとAPUとが識別され、このPEとAPUとからセルが起動し、必要な場合に、宛先PEとAPUとがセルに関する追加情報を得ることが可能となる。応答ID2314にはネットワーク・アドレスが含まれ、この応答ID2314によって、セルに関するクエリとセルの処理の結果とを送る送り先のPEとAPUとが識別される。

【0093】セルの本体部分2306にはネットワークのプロトコルとは無関係の情報が含まれる。図41の分解部分はセルの本体部分2306の細部を図示する。セルの本体部分2306のヘッダ2320によってセル本体の開始部が識別される。セル・インターフェース2322にはセルの利用に必要な情報が含まれる。この情報の中には、グローバルな一意のID2324と、要求されるAPU2326と、サンドボックス・サイズ2328と、前回のセルのID2330とが含まれる。

【0094】グローバルな一意のID2324によって、ネットワーク104全体を通じてソフトウェア・セル2302が一意的に識別される。グローバルな一意のID2324が、ソースID2312(ソースID2312内のPEまたはAPUの一意的識別子など)と、ソフトウェア・セル2302の作成または伝送の時刻と日付とに基づいて作成される。必要なAPU2326によってセルの実行に必要な最低数のAPUと与えられる。サンドボックス・サイズ2328によって、セルの実行に必要なDRAMと関連する必要なAPU内に、保護されたメモリが与えられる。前回のセルID2330によって、シーケンシャルな実行を要求する1グループのセル(ストリーミング・データなど)内の前回のセルの識別子が提供される。

【0095】実行セクション2332の中にはセルのコア情報が含まれる。この情報の中にはDMAコマンド・リスト2334と、プログラム2336と、データ2338とが含まれる。プログラム2336には、APUプログラム2360と2362などのAPUによって実行されるプログラム(“アプレット”と呼ばれる)が含ま

(19)

特開2002-342165

35

れ、データ2338にはこれらのプログラムを用いて処理されるデータが含まれる。DMAコマンド・リスト2334には、プログラムの起動に必要な一連のDMAコマンドが含まれる。これらのDMAコマンドにはDMAコマンド2340、2350、2355、2358が含まれる。PUはDMACへこれらのDMAコマンドを出す。

【0096】DMAコマンド2340にはVID2342が含まれる。VID2342は、DMAコマンドが出されたとき物理IDに対して対応づけられるAPUのバーチャルIDである。DMAコマンド2340にはロード・コマンド2344とアドレス2346も含まれる。ロード・コマンド2344は、APUにDRAMから特定の情報を読み出しローカル・ストレージの中へ入れるように命令する。アドレス2346によってこの特定情報を含むDRAM内のバーチャル・アドレスが与えられる。この特定情報は、プログラム・セクション2336からのプログラムや、データ・セクション2338からのデータや、あるいはその他のデータなどであってもよい。最終的に、DMAコマンド2340にはローカル・ストレージのアドレス2348が含まれる。このアドレスによって、情報をロードできそうなローカル・ストレージのアドレスが識別される。DMAコマンド2350には類似の情報が含まれる。その他のDMAコマンドも使用可能である。

【0097】DMAコマンド・リスト2334には一連のキック・コマンド(キック・コマンド2355と2358など)も含まれる。キック・コマンドとは、PUによってAPUへ出されるセルの処理を開始するコマンドである。DMAキック・コマンド2355には、バーチャルAPU ID2352と、キック・コマンド2354と、プログラム・カウンタ2356とが含まれる。バーチャルAPU ID2352はキックすべき対象APUを識別し、キック・コマンド2354は関連するキック・コマンドを与え、プログラム・カウンタ2356は、プログラムの実行用プログラム・カウンタのためのアドレスを与える。DMAキック・コマンド2358は、同じAPUまたは別のAPUに対して同様の情報を与える。

【0098】上述したように、PUは独立したプロセッサとしてAPUを扱い、コプロセッサとして扱うものではない。したがって、APUによる処理を制御するために、PUは、遠隔手動呼出しに類似したコマンドを使用する。これらのコマンドは“APU遠隔手動呼出し(ARPC)”と呼ばれる。PUは、一連のDMAコマンドをDMACへ出すことによりARPCを実行する。DMACは、APUプログラムとそれに関連するスタック・フレームとをAPUのローカル・ストレージの中へロードする。次いで、PUはAPUへ最初のキックを出し、APUプログラムを実行する。

36

【0099】図42は、アプレットを実行するためのARPCのステップを例示する。指定APUによるアプレットの処理の開始時にPUが実行するこれらのステップが、図42の第1の部分2402に示され、指定APUが実行するステップが、図42の第2の部分2404に示されている。

【0100】ステップ2410で、PUはアプレットを評価し、次いで、アプレットの処理用APUを指定する。ステップ2412で、PUは、必要な単復のサンドボックス用のメモリ・アクセス・キーの設定を行うDMAコマンドをDMACへ出すことにより、アプレットの実行用スペースをDRAM内に割り振る。ステップ2414で、PUは、指定APUへの割り込み要求による、アプレットの完了信号の伝送を可能にする。ステップ2418で、PUは、DRAMからAPUのローカル・ストレージへアプレットをロードするDMAコマンドをDMACへ出す。ステップ2420で、DMAコマンドが実行され、アプレットがDRAMからローカル・ストレージへ読み出される。ステップ2422で、PUは、アプレットと関連付けられたスタック・フレームをDRAMからAPUのローカル・ストレージへロードするDMAコマンドをDMACへ出す。ステップ2423で、DMAコマンドが実行され、スタック・フレームがDRAMからAPUのローカル・ストレージへ読み出される。ステップ2424で、PUは、DMACがAPUへキーを割り当てて、ステップ2412で指定された、一又は複数のハードウェア・サンドボックスからのデータ読み出しと、その一又は複数のハードウェア・サンドボックスへのデータ書き込みを行うことをAPUに許可するDMAコマンドを出す。ステップ2426で、DMACは、APUへ割り当てられたキーを用いてキー管理テーブル(KTAB)の更新を行う。ステップ2428で、PUは、プログラムの処理を開始するDMAコマンド“キック”をAPUへ出す。特定のアプレットに応じて、特定のARPCの実行時にPUによって他のDMAコマンドを出してもよい。

【0101】上記のように、図42の第2の部分2404は、アプレットの実行時にAPUによって行われるステップを例示するものである。ステップ2430で、APUは、ステップ2428で出されるキック・コマンドに応じてアプレットの実行を開始する。ステップ2432で、アプレットの指示で、APUは、アプレットの関連スタック・フレームの評価を行う。ステップ2434で、APUは、DMACへ複数のDMAコマンドを出し、スタック・フレームが必要に応じてDRAMからAPUのローカル・ストレージへ指定するデータのロードを行う。ステップ2436で、これらのDMAコマンドが実行され、データは、DRAMからAPUのローカル・ストレージへ読み出される。ステップ2438でAPUはアプレットを実行し、ある結果を出力する。ステッ

(20)

特開2002-342165

37

38

ブ2440で、APUはDMACへDMAコマンドを出し、DRAMにその結果を格納する。ステップ2442で、DMAコマンドが実行され、アプレットの結果がAPUのローカル・ストレージからDRAMへ書き込まれる。ステップ2444で、APUはPUへ割込み要求を出し、APPCが完了したことを示す信号伝送を行う。

【0102】PUの指示の下で独立にタスクを実行するAPUの能力によって、1グループのAPUと、1グループのAPUと関連付けられたメモリ・リソースとを拡張タスクの実行専用にすることが可能になる。例えば、1つのPUは、1以上のAPUと、これらの1以上のAPUと関連付けられた1グループのメモリ・サンドボックスとを、拡張された時間中ネットワーク104を介して伝送されてくるデータの受信専用とし、また、1以上の他のAPUとそれらと関連付けられたメモリ・サンドボックスへ、この時間中受信したデータのさらなる処理を行うための送信専用とすることができる。この能力は、ネットワーク104を介して伝送されるストリーミング・データ(ストリーミングMPEGまたはストリーミングATRACオーディオまたはビデオ・データなど)の処理にとって特に好適である。PUは、1以上のAPUおよびそれらと関連付けられたメモリ・サンドボックスをこれらのデータの受信専用とし、1以上の他のAPUおよびそれらと関連付けられたメモリ・サンドボックスをこれらのデータの解凍と処理専用とすることができる。言い換えれば、PUは、APUのグループとそれらと関連付けられたメモリ・サンドボックスとの間でこのようなデータ処理を行うための専用パイプライン関係の確立を行うことができる。

【0103】しかし、このような処理を効率的に実行するためには、パイプ・ラインの専用APUとメモリ・サンドボックスとが、データ・ストリームを含むアプレットの処理が行われない時間中もパイプ・ライン専用のままであることが望ましい。言い換えれば、専用APUおよびそれらと関連するサンドボックスが、これらの時間中予約状態のままに置かれることが望ましい。アプレットの処理の完了時における、APUとその関連付けられた一又は複数のメモリ・サンドボックスを予約、即ちリザーブ状態としておくことは、“常駐終了”と呼ばれる。常駐終了はPUからの命令に応じて行われる。

【0104】図43、44、45は、1グループのAPUおよびそれらと関連するサンドボックスを含む、ストリーミング・データ(ストリーミングMPEGデータなど)を処理するための専用パイプライン構造の設定を例示する。図43に示すように、このパイプライン構造の構成要素にはPE2502とDRAM2518とが含まれる。PE2502の中には、PU2504、DMAC2506およびAPU2508、APU2510、APU2512を含む複数のAPUが含まれる。PU2504、DMAC2506およびこれらのAPU間の通信は

PEバス2514を介して行われる。広帯域幅のバス2516によってDMAC2506はDRAM2518と接続される。DRAM2518の中には、複数のサンドボックス(サンドボックス2520、サンドボックス2522、サンドボックス2524、サンドボックス2526など)が含まれる。

【0105】図44に、専用パイプラインを設定するためのステップを例示する。ステップ2610で、PU2504は、ネットワーク・アプレットを処理するようにAPU2508を割り当てる。ネットワーク・アプレットは、ネットワーク104のネットワーク・プロトコルの処理用プログラムを有する。この場合、このプロトコルは伝送制御プロトコル/インターネット用プロトコル(TCP/IP)である。このプロトコルに従うTCP/IPデータ・パケットはネットワーク104を介して伝送される。受信時に、APU2508はこれらのパケットを処理し、パケット内のデータを組み立て、ソフトウェア・セル102の中へ入れる。ステップ2612で、PU2504は、ネットワーク・アプレットの処理の完了時に常駐終了を実行するようにAPU2508に指示する。ステップ2614で、PU2504は、APU2510及び2512がMPEGアプレットの処理を行うように割り当てる。ステップ2615で、PU2504は、MPEGアプレットの処理の完了時に常駐終了を実行するようにAPU2510及び2512に指示する。ステップ2616で、PU2504は、APU2508とAPU2510によるアクセス用ソース・サンドボックスとしてサンドボックス2520を指定する。ステップ2618で、PU2504は、APU2510によるアクセス用宛先サンドボックスとしてサンドボックス2522を指定する。ステップ2620で、PU2504は、APU2508とAPU2512によるアクセス用ソース・サンドボックスとしてサンドボックス2524を指定する。ステップ2622で、PU2504は、APU2512によるアクセス用宛先サンドボックスとしてサンドボックス2526を指定する。ステップ2624で、APU2510とAPU2512とは、それぞれ、ソース・サンドボックス2520とソース・サンドボックス2524の範囲内のメモリ・ブロックへ同期読取りコマンドを送り、これらのメモリ・ブロックをブロッキング状態に設定する。最後に、処理はステップ2628へ移り、そこで、専用パイプラインの設定が完了し、パイプ・ライン専用のリソースが予約される。このようにして、APU2508、2510、2512およびそれらと関連するサンドボックス2520、2522、2524および2526は予約状態に入る。

【0106】図45に、この専用パイプラインによるストリーミングMPEGデータの処理ステップを例示する。ステップ2630で、APU2508は、ネットワーク・アプレットを処理し、そのローカル・ストレージ

(21)

特開2002-342165

39

の中で、TCP/IPデータ・パケットをネットワーク104から受信する。ステップ2632で、APU2508は、これらのTCP/IPデータ・パケットを処理し、これらのパケット内のデータをアセンブルし、ソフトウェア・セル102の中へ入れる。ステップ2634で、APU2508はソフトウェア・セルのヘッダ2320(図23)をチェックし、セルがMPEGデータを含むかどうかの判定を行う。セルがMPEGデータを含まない場合、ステップ2636で、APU2508は、専用パイプライン内に含まれない他のAPUによって他のデータ処理するために、DRAM2518内に指定される汎用サンドボックスへそのセルを伝送する。またAPU2508はこの伝送についてPU2504に通知する。

【0107】一方、ソフトウェア・セルがMPEGデータを含む場合、ステップ2638で、APU2508はそのセルの前のセルのID2330(図23)をチェックし、そのセルが属するMPEGデータ・ストリームを識別する。ステップ2640で、APU2508はセルの処理用の専用パイプラインのAPUを選択する。この場合、APU2508は、これらのデータを処理するAPU2510を選択する。この選択は前回のセルID2330とロード・バランシング・ファクタ(負荷平衡係数)とに基づく。例えば、そのソフトウェア・セルが属するMPEGデータ・ストリームの前回のソフトウェア・セルが処理用としてAPU2510へ送られたことが前のセルID2330によって示されている場合、現在のソフトウェア・セルも通常の処理用としてAPU2510へ送られる。ステップ2642で、APU2508は、サンドボックス2520へMPEGデータを書き込む同期書き込みコマンドを出す。このサンドボックスは予めブロッキング状態に設定されているので、ステップ2644で、MPEGデータは、サンドボックス2520からAPU2510のローカル・ストレージへ自動的に読み出される。ステップ2646で、APU2510はそのローカル・ストレージでMPEGデータを処理してビデオ・データを生成する。ステップ2648で、APU2510はサンドボックス2522へビデオ・データを書き込む。ステップ2650で、APU2510はサンドボックス2520へ同期読取りコマンドを出し、このサンドボックスに追加MPEGデータの受信を準備させる。ステップ2652で、APU2510は常駐終了処理を行う。この処理によってこのAPUは予約状態に入り、この予約状態の間APUは、MPEGデータ・ストリームの中で追加MPEGデータの処理を行うべく待機する。

【0108】他のタイプのデータ処理用として1グループのAPUおよびそれらに関連するサンドボックス間でその他の専用構造の設定が可能である。例えば、図46に示すように、APUの専用グループ(APU270

40

2、2708、2714など)を設定し、3次元オブジェクトに対して幾何学的変換を実行して2次元ディスプレイ・リストの生成を行うことが可能となる。これらの2次元ディスプレイ・リストを他のAPUによってさらに処理(レンダ)し画素データの生成を行うようにすることが可能である。この処理を実行するために、3次元オブジェクトと、これらのオブジェクト処理から結果として生じるディスプレイ・リストの格納用として、サンドボックスが、APU2702、2708、2414の専用となる。例えば、ソース・サンドボックス2704、2710、2716は、それぞれ、APU2702、APU2708、APU2714によって処理された3次元オブジェクトの格納専用となる。同様に、宛先サンドボックス2706、2712、2718は、それぞれ、APU2702、APU2708、APU2714によるこれらの3次元オブジェクトの処理から結果として生じるディスプレイ・リストの格納専用となる。

【0109】調整用APU2720は、そのローカル・ストレージにおける、宛先サンドボックス2706、2712、2718からのディスプレイ・リストの受信専用である。APU2720は、これらのディスプレイ・リスト間での調整を行い、画素データのレンダリングのためにこれらのディスプレイ・リストを他のAPUへ送る。

【0110】システム101のプロセッサは絶対タイマーも使用する。この絶対タイマーはAPUとPEの他のエレメントへクロック信号を出力する。このクロック信号はこれらのエレメントを駆動するクロック信号に依存せず、かつ、このクロック信号より高周波である。この絶対タイマーの利用が図28に例示されている。

【0111】この図に示すように、この絶対タイマーによってAPUによるタスク・パフォーマンスのためのタイム・バジェット(割り当て時間)が決定される。このタイム・バジェットによって、これらのタスクの完了時間が設定されるが、この時間はAPUによるタスク処理に必要な時間より長い時間になる。その結果、各タスクについて、タイム・バジェットの範囲内に、ビジー時間とスタンバイ時間とが存在することになる。すべてのアプレットは、APUの実際の処理時間にかかわらず、このタイム・バジェットに基づいて処理を行うように言われる。

【0112】例えば、PEの特定のAPU用として、タイム・バジェット2804のビジー時間2802中に特定のタスクを行うことができる。ビジー時間2802がタイム・バジェット2804未満であるため、スタンバイ時間2806がタイム・バジェット中に生じる。このスタンバイ時間中、APUは、APUが消費するパワーが少なくなるスリープモードに入る。

【0113】タイム・バジェット2804が満了するまで、他のAPUまたはPEの他のエレメントがタス

41

ク処理の結果を予想することはない。したがって、APUの実際の処理速度にかかわらず、絶対タイマーによって決定されるタイム・バジェットを用いてAPUの処理結果が常時調整される。

【0114】将来、APUによる処理速度はさらに高速になる。しかし、絶対タイマーによって設定されるタイム・バジェットは同じままである。例えば、図28に示すように、将来のAPUは、さらに短時間でタスクを実行することになり、したがって、スタンバイ時間はさらに長くなるであろう。したがって、ビジー時間2808はビジー時間2802より短くなり、スタンバイ時間2810はスタンバイ時間2806より長くなる。しかし、絶対タイマーによって設定された同じタイム・バジェットに基づいて処理を行うようにプログラムが書かれているので、APU間での処理結果の調整が維持される。その結果、さらに高速のAPUが、その処理の結果が予想される時点でコンフリクトを生じることなく、低速のAPU用として言われたプログラムの処理を行うことが可能となる。

【0115】動作速度の向上や動作速度が異なることに起因するAPUの並列処理の調整問題に対しては、APU間での調整を決定する絶対タイマーに代えて、PUまたは1以上の指定APUにおいて、APUが実行している特定の命令(マイクロコード)の分析をアプレットの処理時に行うようにすることもできる。“オペレーションなし”(“NOOP”)命令を命令の中へ挿入し、APUのいくつかによってこの命令を実行して、アプレットによって予想されるAPUによる処理を1ステップずつ適切に行うことが可能となる。命令の中へこれらのNOOPを挿入することにより、すべての命令のAPUによる実行を行うための正しいタイミングの維持が可能となる。

【0116】以上特定の実施形態に関して本明細書で本発明について説明したが、これらの実施形態は本発明の原理と適用を示す単に例示的なものであると理解すべきである。したがって、添付の請求項によって画定されているような本発明の精神と範囲から逸脱することなく、以上の例示の実施形態に対して多数の改変を行うことが可能であり、また、他の構成を考案することが可能である。

【図面の簡単な説明】

【図1】 本発明によるコンピュータ・ネットワークのアーキテクチャ全体を例示する。

【図2】 本発明によるプロセッサ・エレメント(PE)の構造を例示する図である。

【図3】 本発明による広帯域エンジン(BE)の構造を例示する図である。

【図4】 本発明による付加処理ユニット(APU)の構造を例示する図である。

【図5】 本発明によるプロセッサ・エレメントと、ビ

(22)

特開2002-342165

42

ジュアライザ(VS)と、光インターフェースとの構造を例示する図である。

【図6】 本発明によるプロセッサ・エレメントの1つの組合せを例示する図である。

【図7】 本発明によるプロセッサ・エレメントの別の組合せを例示する図である。

【図8】 本発明によるプロセッサ・エレメントのさらに別の組合せを例示する図である。

【図9】 本発明によるプロセッサ・エレメントのさらに別の組合せを例示する図である。

【図10】 本発明によるプロセッサ・エレメントのさらに別の組合せを例示する図である。

【図11】 本発明によるチップ・パッケージ内での光インターフェースの統合化を例示する図である。

【図12】 図11の光インターフェースを用いるプロセッサの1つの構成を示す図である。

【図13】 図11の光インターフェースを用いるプロセッサの別の構成を示す図である。

【図14】 本発明によるメモリ・システムの構造を例示する図である。

【図15】 本発明による第1の広帯域エンジンから第2の広帯域エンジンへのデータの書き込みを例示する図である。

【図16】 本発明によるプロセッサ・エレメントのための共用メモリの構造を示す図である。

【図17】 図16に示すメモリ・バンク用の1つの構造を例示する図である。

【図18】 図16に示すメモリ・バンク用の別の構造を例示する図である。

【図19】 本発明によるDMACのための構造を例示する図である。

【図20】 本発明によるDMACのための代替の構造を例示する図である。

【図21】 本発明によるデータ同期オペレーションを例示する図である。

【図22】 本発明によるデータ同期オペレーションを例示する図である。

【図23】 本発明によるデータ同期オペレーションを例示する図である。

【図24】 本発明によるデータ同期オペレーションを例示する図である。

【図25】 本発明によるデータ同期オペレーションを例示する図である。

【図26】 本発明によるデータ同期オペレーションを例示する図である。

【図27】 本発明によるデータ同期オペレーションを例示する図である。

【図28】 本発明によるデータ同期オペレーションを例示する図である。

【図29】 本発明によるデータ同期オペレーションを

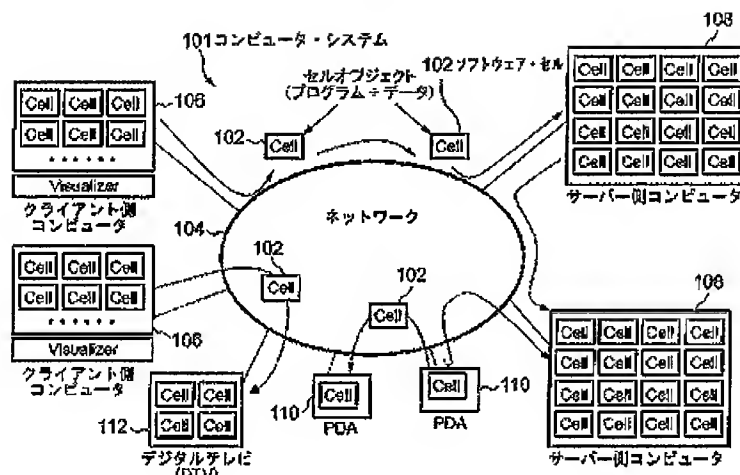
(23)		特開2002-342165	
43		44	
例示する図である。	106	クライアント	
【図30】 本発明によるデータ同期オペレーションを	108	サーバーコンピュータ	
例示する図である。	1104	光インターフェース	
【図31】 本発明によるデータ同期オペレーションを	1108	バス	
例示する図である。	1118	ポート	
【図32】 本発明によるデータ同期オペレーションを	1122	ポート	
例示する図である。	1126	光導波路	
【図33】 本発明によるデータ同期オペレーションを	1160	光インターフェース	
例示する図である。	1162	光インターフェース	
【図34】 本発明によるデータ同期オペレーションを	1164	光インターフェース	
例示する図である。	1166	光インターフェース	
【図35】 本発明によるデータ同期オペレーションを	1182	光インターフェース	
例示する図である。	1184	光インターフェース	
【図36】 本発明のデータ同期方式によるメモリ・ロ	1186	光インターフェース	
ケーションの様々な状態を例示する3つの状態のメモリ	1188	光インターフェース	
図である。	1188	光インターフェース	
【図37】 本発明によるハードウェア・サンドボック	1190	光インターフェース	
ス用のキー管理テーブルの構造を例示する図である。	1190	光インターフェース	
【図38】 本発明によるハードウェア・サンドボック	1206	コントロール	
ス用メモリ・アクセス・キーの格納方式を例示する図で	1212	ユニット	
ある。	1221	クロスバ交換機	
【図39】 本発明によるハードウェア・サンドボック	1232	外部ポート	
ス用メモリ・アクセス管理テーブルの構造を例示する図	1234	コントロール	
である。	1240	ユニット	
【図40】 図37のキー管理テーブルと図39のメモ	1242	コントロール	
リ・アクセス管理テーブルとを用いてメモリ・サンドボ	1244	バンク	
ックスにアクセスするステップを示すフロー・チャート	1406	ブロック	
である。	1414	バンク	
【図41】 本発明によるソフトウェア・セルの構造を	1416	バンク	
例示する図である。	1504	ノード	
【図42】 本発明による、APUへ遠隔処理命令を出	1607	バス	
すステップを示すフロー・チャートである。	1608	バス	
【図43】 本発明による、ストリーミング・データ処	1722	制御回路	
理専用パイプラインの構造を例示する図である。	1724	制御論理回路	
【図44】 本発明によるストリーミング・データの処	1726	ストレージ	
理時の図43の専用パイプラインによって実行されるス	1728	ロケーション	
テップを示すフロー・チャートである。	1729	セグメント	
【図45】 本発明によるストリーミング・データの処	1731	ロケーション	
理時の図43の専用パイプラインによって実行されるス	1732	ロケーション	
テップを示すフロー・チャートである。	1742	制御論理回路	
【図46】 本発明によるストリーミング・データ処理	1746	ロケーション	
用の専用パイプラインの他の構造を例示する図である。	1750	ロケーション	
【図47】 本発明によるAPUによるアプリケーション	1752	セグメント	
とデータの並列処理を調整するための絶対タイマー方	1760	セグメント	
式を例示する図である。	1762	セグメント	
【符号の説明】	1880	エンブティ状態	
101 システム	1882	フル状態	
1010 キー	1884	ブロッキング状態	
102 セル	1902	キー管理テーブル	
104 ネットワーク	1906	キー	

(24)

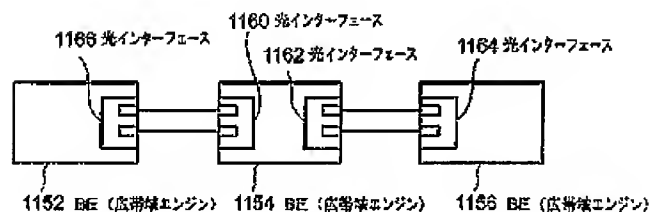
特開2002-342165

45		46	
1908	マスク	* 2524	サンドボックス
2006	格納位置	2526	サンドボックス
2008	セグメント	2704	サンドボックス
2010	セグメント	2706	宛先サンドボックス
2012	キー	301	広帯域エンジン
2102	アクセス管理テーブル	311	バス
2106	アドレス	313	広帯域メモリ接続部
2110	キー	317	インターフェース
2110	キーマスク	319	外部バス
223	バス	10 406	メモリ
227	高帯域メモリ接続部	408	内部バス
2302	セル	410	レジスタ
2308	ヘッダ	412	浮動小数点演算ユニット
2320	ヘッダ	414	整数演算ユニット
2322	インターフェース	420	バス
2332	実行セクション	506	パッケージの中に光インターフェース
2334	リスト	508	エンジン
2520	サンドボックス	510	画像用キャッシュ
2522	サンドボックス	*	

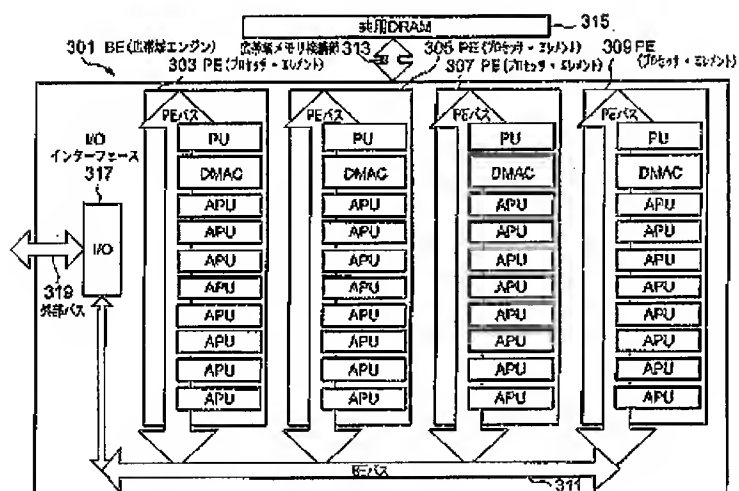
【図1】



【図12】



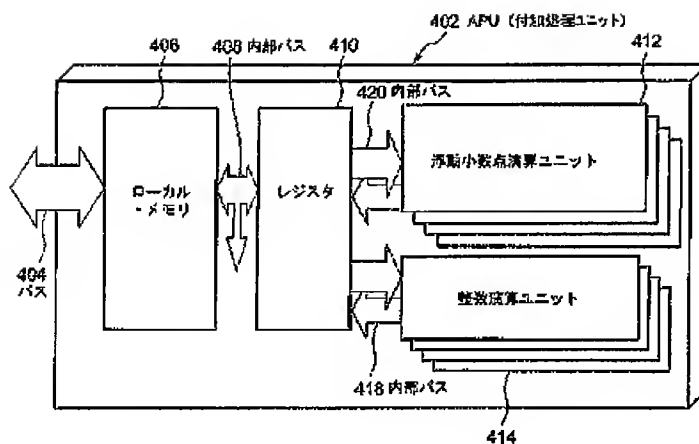
【圖2】



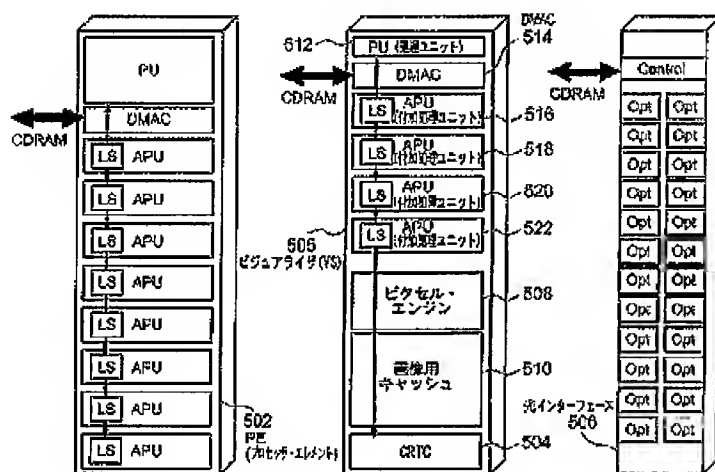
(26)

特開2002-342165

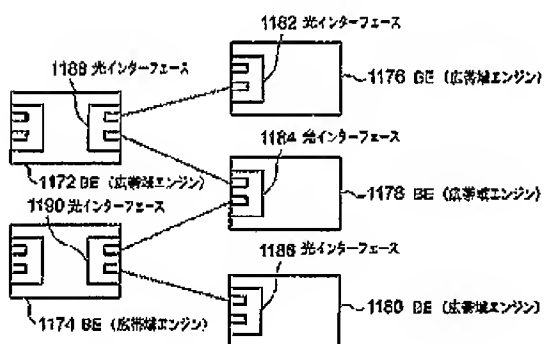
【図4】



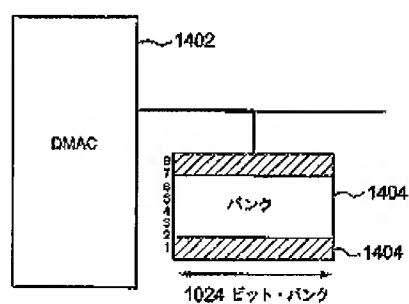
【図5】



【図13】



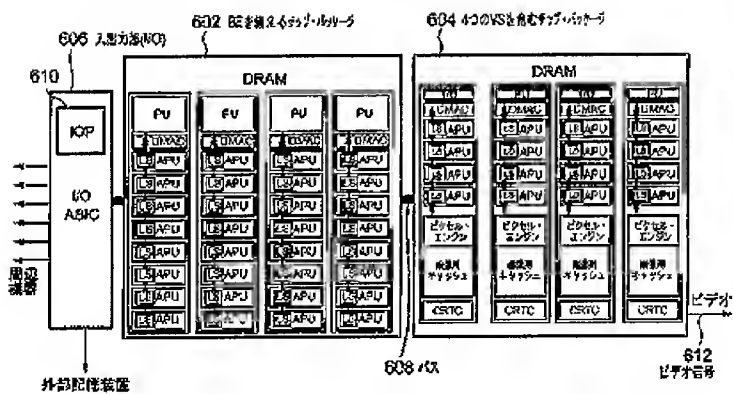
【図17】



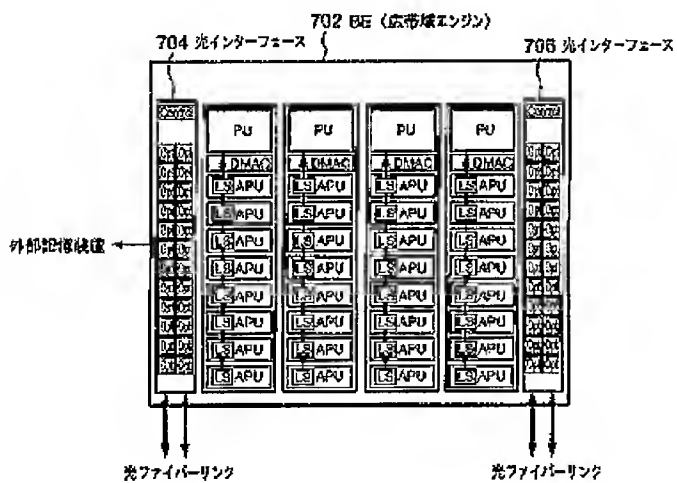
(27)

特開2002-342165

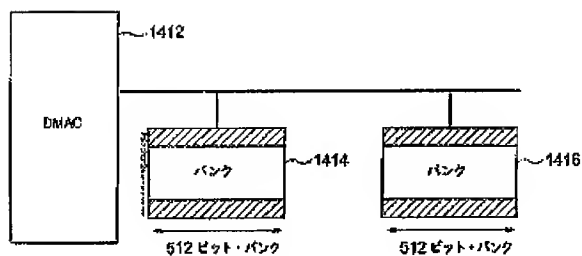
【图6】



【圖 7】



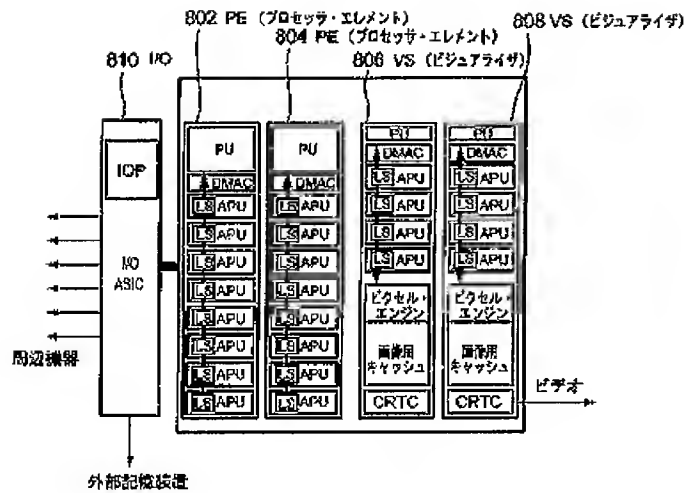
【圖 18】



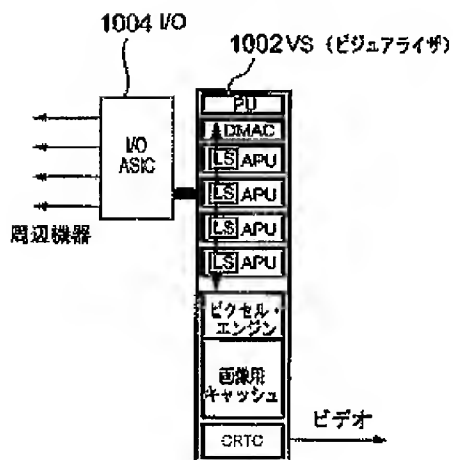
(28)

特開2002-342165

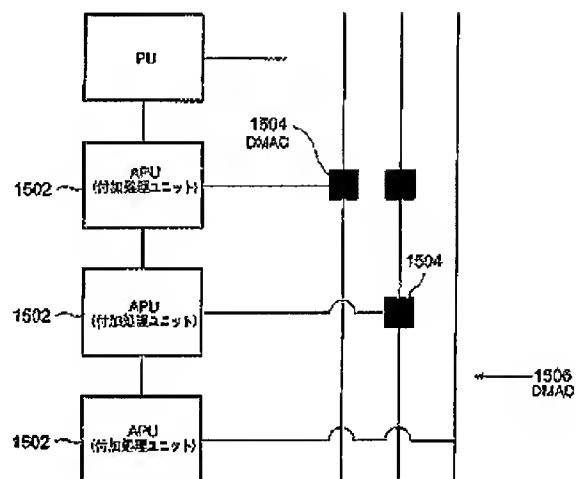
【図8】



【図10】



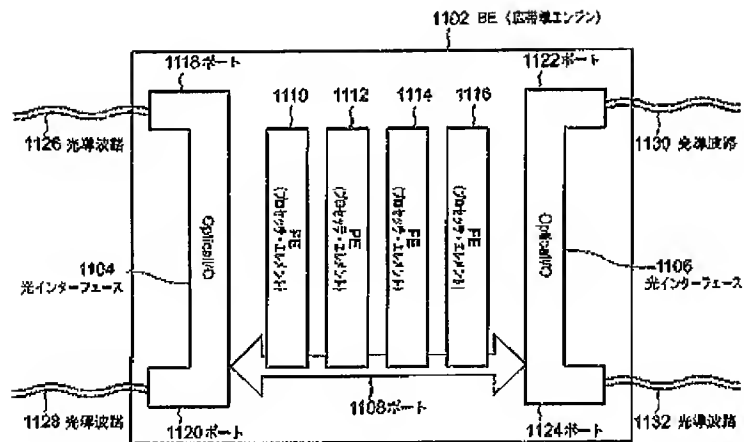
【図19】



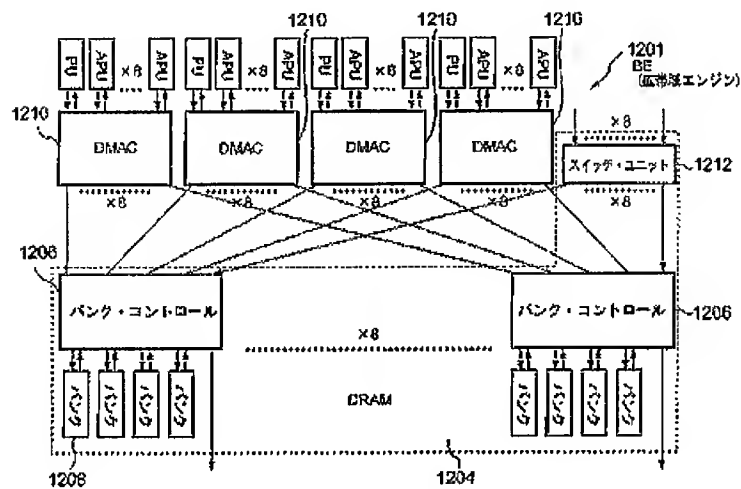
(29)

特開2002-342165

【図11】



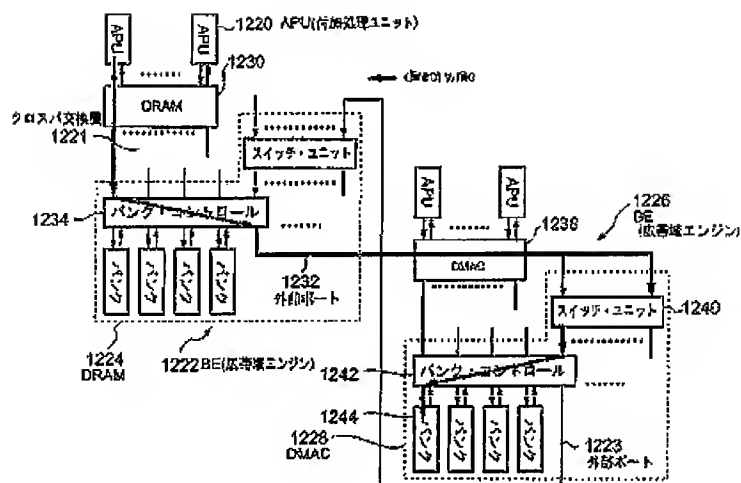
【図14】



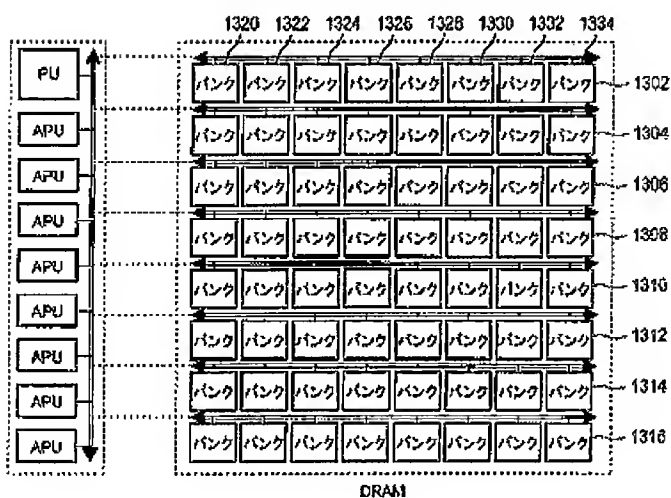
(30)

特開2002-342165

【図15】



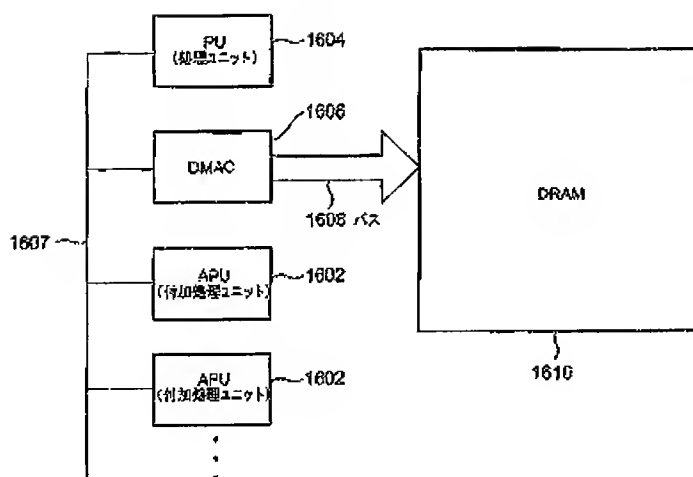
【図16】



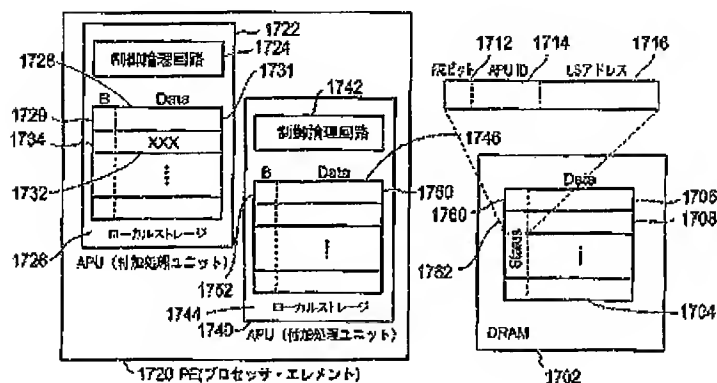
(31)

特開2002-342165

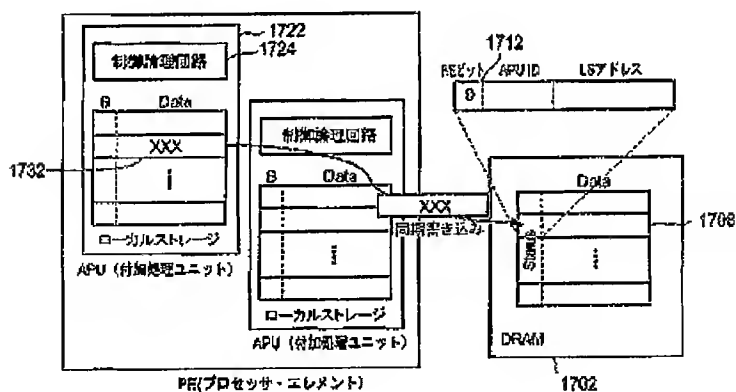
【図20】



【図21】



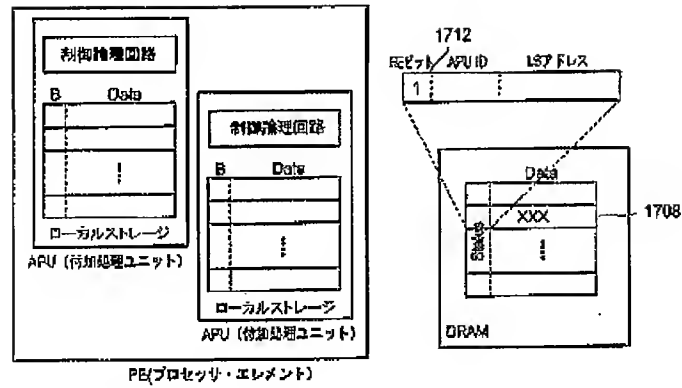
【図22】



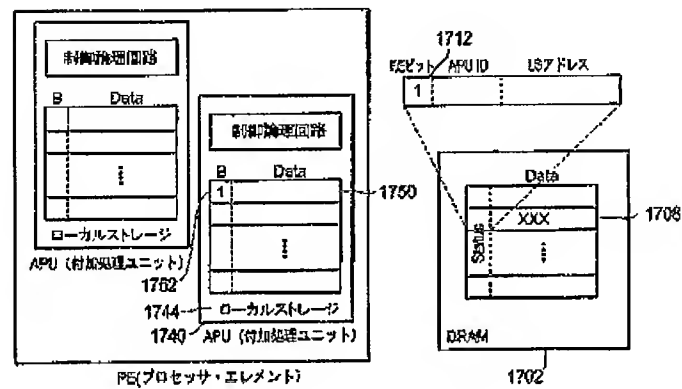
(32)

特開2002-342165

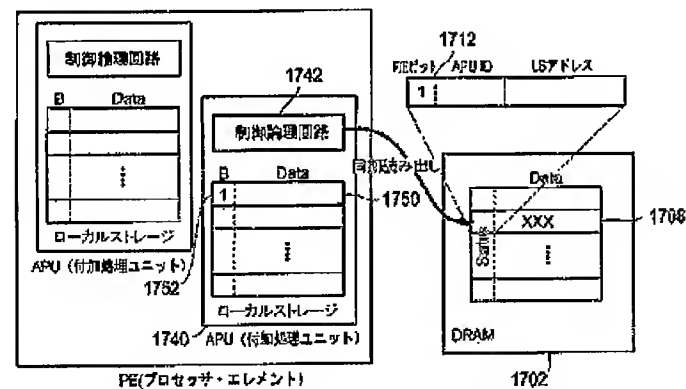
【図23】



【図24】



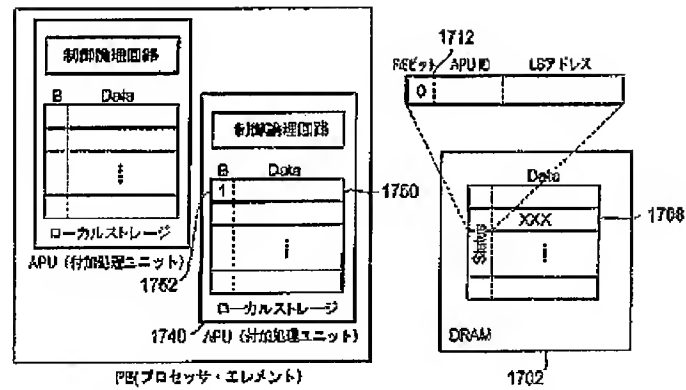
【図25】



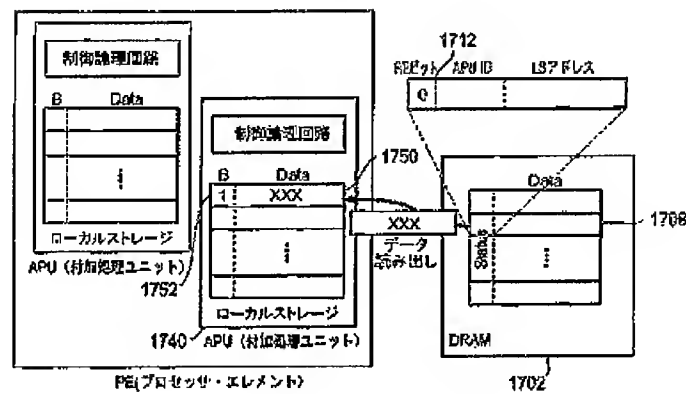
(33)

特開2002-342165

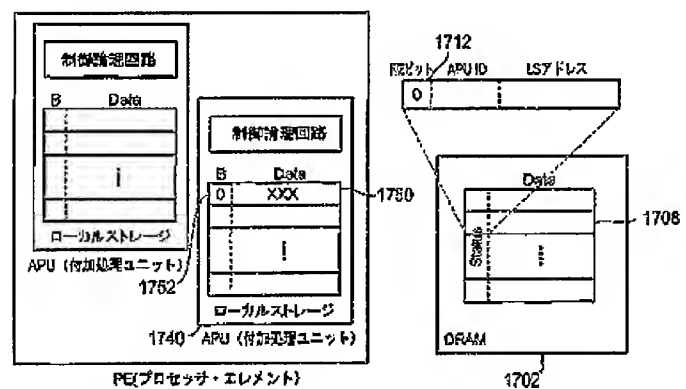
【図26】



【図27】



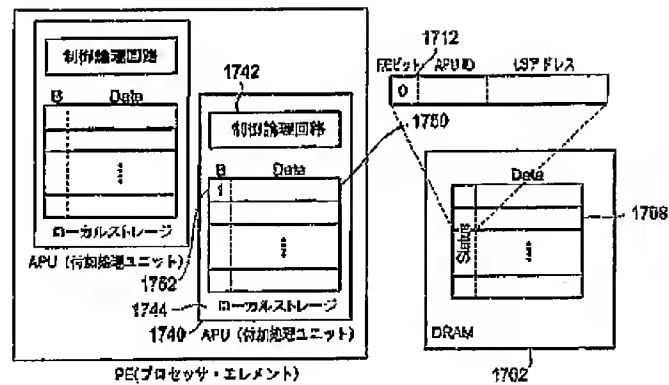
【図28】



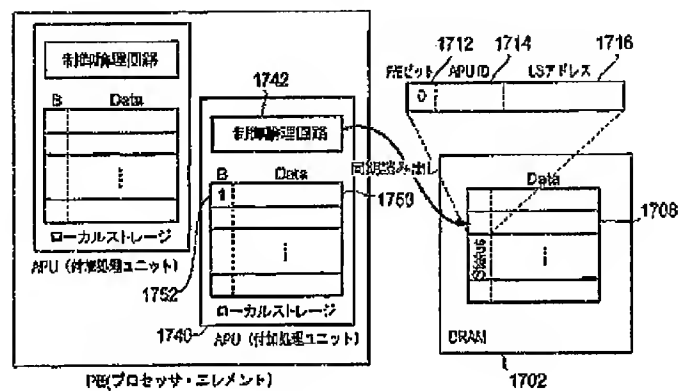
(34)

特開2002-342165

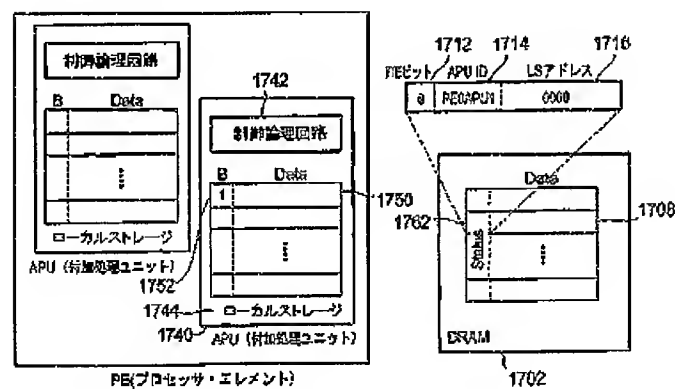
【図29】



【図30】



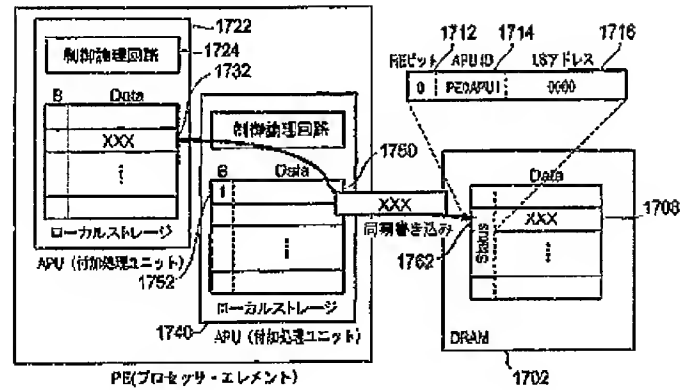
【図31】



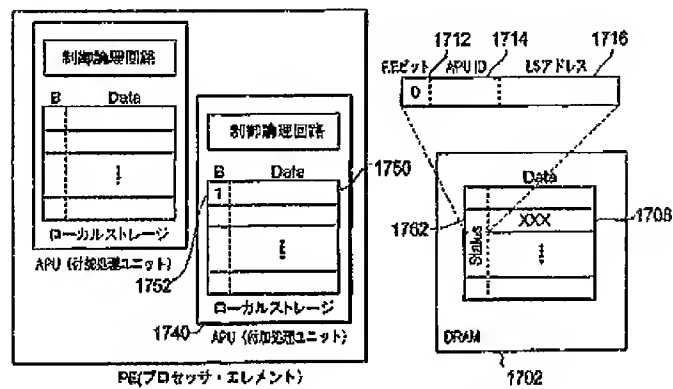
(35)

特開2002-342165

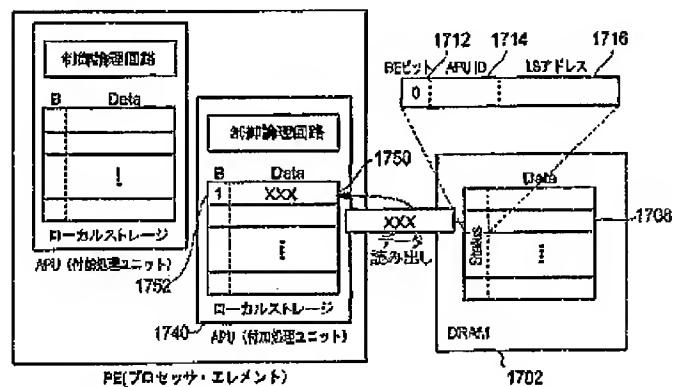
【図32】



【図33】

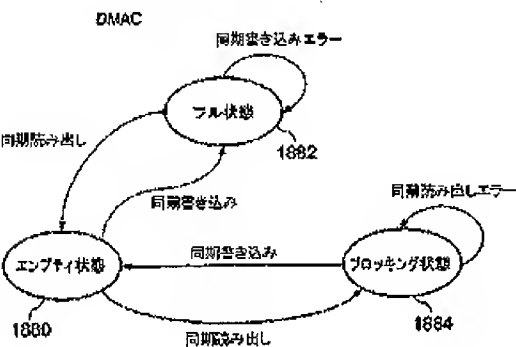


【図34】

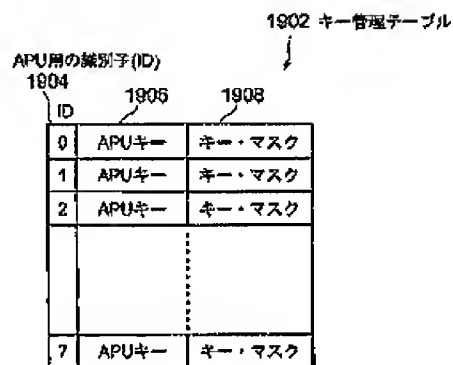


特開2002-342165

【图 38】



【图 38】



(37)

特開2002-342165

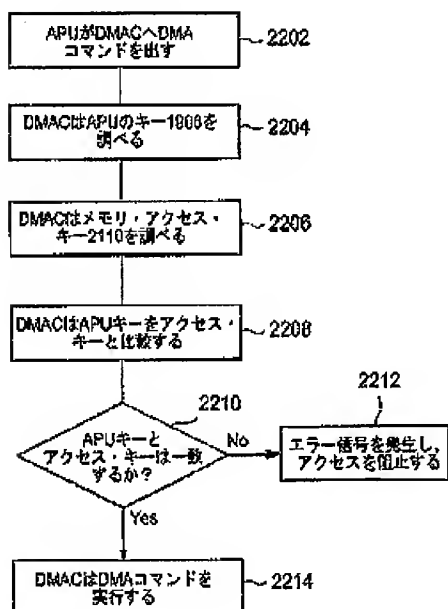
【図39】

2102 メモリ・アクセス管理テーブル

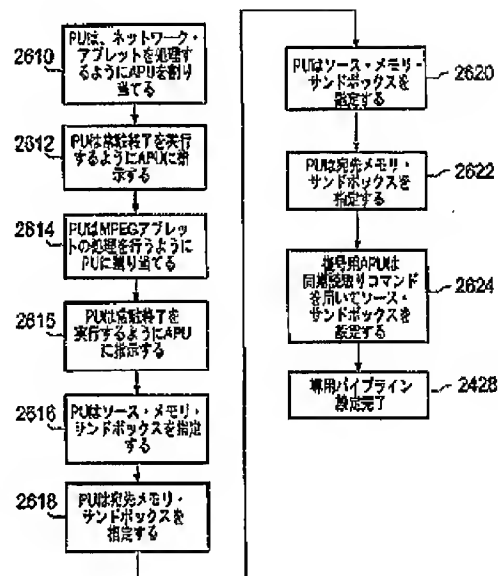
ID	ベース・メモリ・アドレス 2106	サンドボックス・サイズ 2108	アクセス・キーマスク/メモリ・アクセスキー 2110	2112 ベース・メモリ・アドレス
0	Base	Size	Access Key	Access Key Mask
1	Base	Size	Access Key	Access Key Mask
2	Base	Size	Access Key	Access Key Mask
		⋮		
63	Base	Size	Access Key	Access Key Mask

2104 サンドボックス識別子(ID)

【図40】



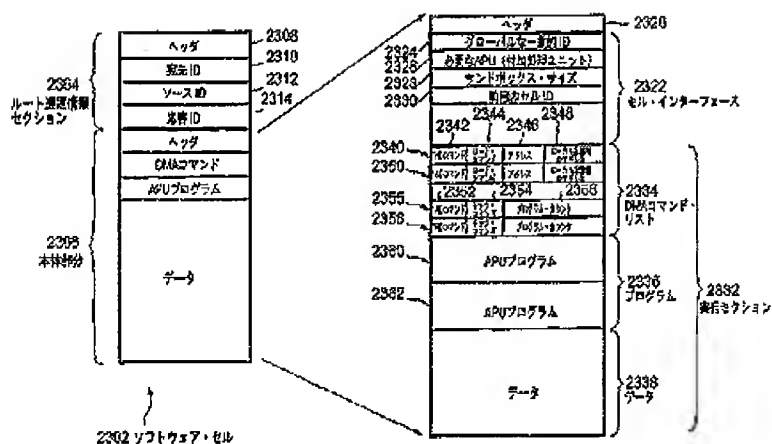
【図44】



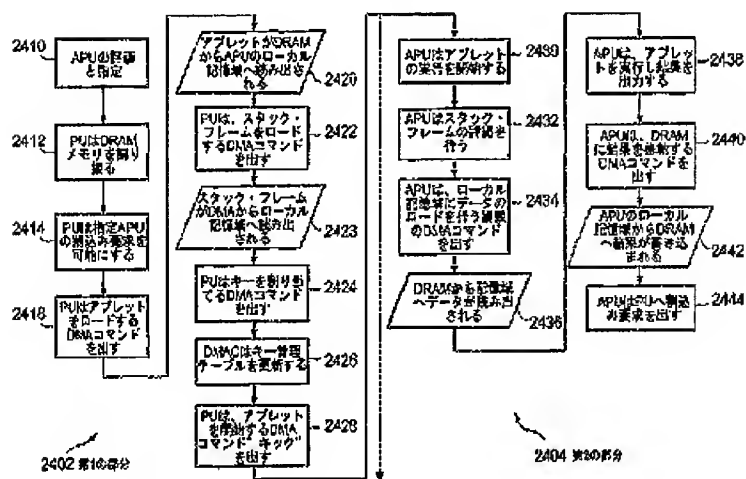
(38)

特開2002-342165

【図41】



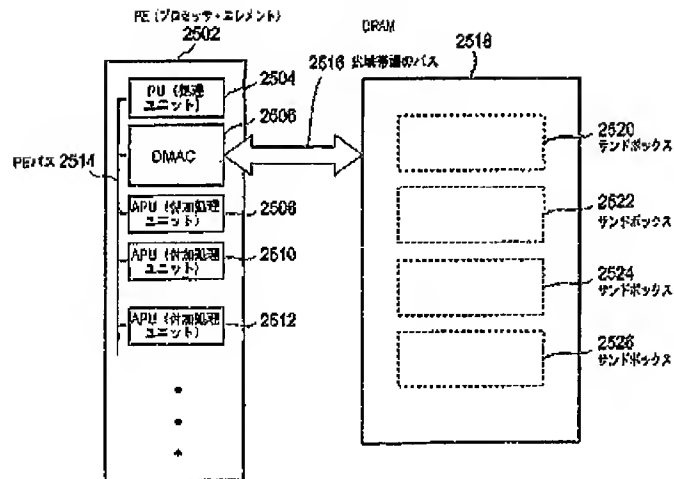
【图 42】



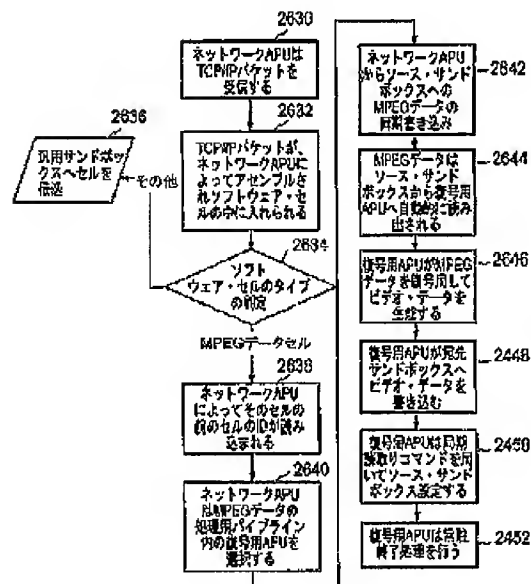
(39)

特開2002-342165

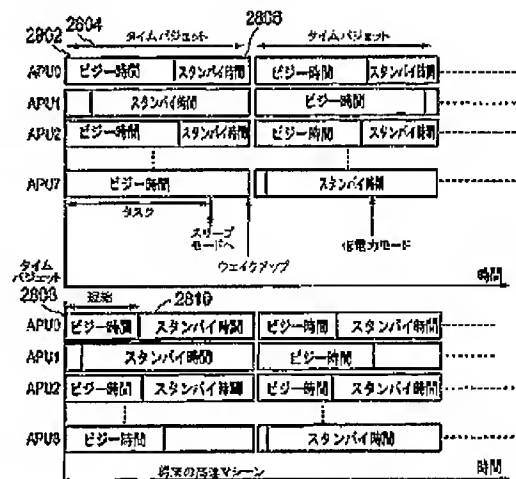
【図43】



【図45】



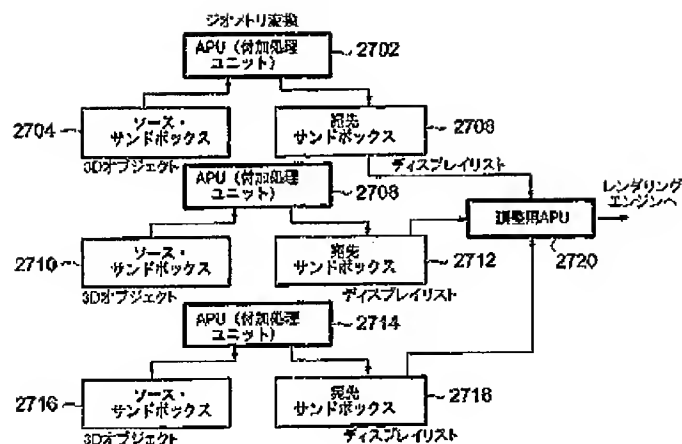
【図47】



(40)

特開2002-342165

【図46】



 フロントページの続き

(72)発明者 山崎 剛

アメリカ合衆国、カリフォルニア州
 94404-2175、フォスター シティ、セ
 カンド フロア、イースト ヒルスデイル
 ブルバード 919 ソニー コンピュー
 タエンタテインメント アメリカ、イン
 ク、内

Fターム(参考) 5B017 AA01 BA02 CA15
 5B045 BB28 GG08 GG11